UNIVERSIdade FEEVALE

MARCOS ENRIQUE SCHMIDT

INTEGRAÇÃO DE SISTEMAS ATRAVÉS DA FERRRAMENTA *WINDOWS SCRIPT HOST*

Novo Hamburgo

2014

MARCOS ENRIQUE SCHMIDT

INTEGRAÇÃO DE SISTEMAS ATRAVÉS DA FERRAMENTA *WINDOWS SCRIPT HOST*

Trabalho de Conclusão de Curso

apresentado como requisito parcial

à obtenção do grau de Bacharel em

Sistemas de Informação pela

Universidade Feevale

Orientador: Juliano Varella de Carvalho

Novo Hamburgo

2014

Agradecimentos

A minha mãe Eda por ser responsável pelas minhas virtudes e por estar sempre ao meu lado, torcendo, dando carinho e servindo de exemplo.

A minha irmã Silvia por simplesmente ser a minha melhor amiga além de ser responsável por duas das grandes alegrias que tenho nesta vida, meus sobrinhos Luna e Ramon.

A minha esposa Patrícia e ao meu filho Henrique, fontes constantes de amor, alegria, inspiração e estímulo.

Por fim, ao meu orientador Juliano e a todos os professores que contribuíram para o meu conhecimento com seus ensinamentos.

Resumo

As empresas tem buscado obter o máximo de retorno possível dos sistemas de informação, isto faz com que ao invés de utilizar um sistema único, passem a utilizar sistemas especialistas para cada área. Como exemplo, podem-se citar os sistemas para RH, produção, custos, governo entre outros. Este cenário torna imprescindível a integração de sistemas a fim de reduzir custos com o retrabalho, erros de digitação, garantir a confiabilidade dos dados e principalmente, que os dados atendam aos critérios definidos, evitando informações inconsistentes. No cenário atual muitos sistemas já possuem ferramentas para possibilitar a integração. Porém, existem determinados sistemas proprietários que não disponibilizam as mesmas. Isto faz com que o usuário tenha que capturar os dados de forma manual, utilizando a exportação de dados e\ou a impressão de relatórios em arquivos. O presente trabalho busca apresentar uma forma de possibilitar a comunicação entre dois sistemas, através da captura de dados via interface. Tal técnica visa aumentar a confiabilidade dos dados e propiciar redução de custos operacionais com a diminuição do tempo utilizado pelo usuário, além de permitir maior rapidez no processo.

Palavras-chave: Integração de sistemas. Interface. *Windows Script Host*. AutoHotkey.

Abstract

The companies have looked for to getting the maximum possible return on information systems, so instead of using a single system, they tend to use specialized systems for each area. As an example, it is quoted the systems to the human resources, production, costs, government, among others. This setting makes the integration of data essential, in order to reduce costs through the redoing, typing mistakes, maintaining the reliability of the data, and mainly that the data ensures the defined criteria, avoiding inconsistent information. In the current setting many systems have already had owners that do not provide these ones. So, the user has to take the data manually, using the export of data and/or the printing of reports into files. This present study aims at presenting a way to enable the communication between the two systems, through the data capture via interface. This technique aims at improving the reliability of the operational costs by the decreasing of time spent for the user and allowing a greater speed in the process.

Key words: Systems integration. Interface. Windows Script Host. AutoHotkey.

Lista de Figuras

[Figura 1 Fases de Implementação 15](#_Toc392351410)

[Figura 2 ODBC - Modelo Funcional 18](#_Toc392351411)

[Figura 3 JDBC Modelo Funcional 19](#_Toc392351412)

[Figura 4 - Integração por troca de mensagens 20](#_Toc392351413)

[Figura 5 Exemplo de RPC 21](#_Toc392351414)

[Figura 6 Funcionamento do WebService 23](#_Toc392351415)

[Figura 7 Programa cliente acessando vários Webservices 23](#_Toc392351416)

[Figura 8 GID Tela de entrada do sistema 27](#_Toc392351417)

[Figura 9 GID Menu para acesso a consulta 27](#_Toc392351418)

[Figura 10 GID Tela para realizar a consulta 28](#_Toc392351419)

[Figura 11 GID Procedimento para exportar os dados 28](#_Toc392351420)

[Figura 12 Dados copiados para o editor de textos 29](#_Toc392351421)

[Figura 13 Sistema CFC - Menu Acesso Crítica GID 29](#_Toc392351422)

[Figura 14 Sistema CFC - Tela para seleção do arquivo a ser importado 30](#_Toc392351423)

[Figura 15 Sistema CFC - Relatório de Crítica de dados não lançados 31](#_Toc392351424)

[Figura 16 Protótipo nova tela de crítica entre os sistemas 32](#_Toc392351425)

[Figura 17 Fluxograma de funcionamento *Script* 33](#_Toc392351426)

[Figura 18 - Histórico de Versões do Windows Script Host 35](#_Toc392351427)

[Figura 19 Arquitetura WSH 36](#_Toc392351428)

[Figura 20 Parâmetros de linha de comando do *CScript* 37](#_Toc392351429)

[Figura 21 Elementos XML que podem ser utilizados em um *script* WSH. 38](#_Toc392351430)

[Figura 22 - Modelo de objeto do *Windows Script Host* 39](#_Toc392351431)

[Figura 23 Processo de crítica atual 48](#_Toc392351432)

[Figura 24 Processo de crítica novo utilizando WSH e AutoHotkey 48](#_Toc392351433)

[Figura 25 Declaração da Referência *Early Binding* 49](#_Toc392351434)

[Figura 26 Declaração *Early Binding* 50](#_Toc392351435)

[Figura 27 Declaração *Late Binding* 50](#_Toc392351436)

[Figura 28 Utilização do mouse para exportar os dados referente as aulas ministradas 52](#_Toc392351437)

[Figura 29 Tela de Parâmetros dos Tempos 61](#_Toc392351438)

[Figura 30 Resultado da integração 62](#_Toc392351439)

Lista de Tabelas

[Tabela 1 Objetos WSH e tarefas associadas 39](#_Toc392351440)

Lista de QUADROS

[Quadro 1 WSH – Exemplo de *Script*  para criar arquivo Excel 42](#_Toc392351441)

[Quadro 2 WSH Exemplo de Consulta no baco de dados Oracle 43](#_Toc392351442)

[Quadro 3 WSH Exemplo de integração entre os programas Excel e Bloco de Notas 45](#_Toc392351443)

[Quadro 4 Declaração API para controlar a posição do cursor 53](#_Toc392351444)

[Quadro 5 Declaração API para controlar os eventod do *mouse* 53](#_Toc392351445)

[Quadro 6 Declaração de variáveis para controle do *mouse* 53](#_Toc392351446)

[Quadro 7 Função para uso botão esquerdo do *mouse* 54](#_Toc392351447)

[Quadro 8 Função para uso botão direito do mouse 54](#_Toc392351448)

[Quadro 9 Função para uso duplo clique do mouse 54](#_Toc392351449)

[Quadro 10 Exemplo de uso das funções SetCursorPos e mouse\_event 54](#_Toc392351450)

[Quadro 11 WSH - Posicionando o Foco no sistema GID 56](#_Toc392351451)

[Quadro 12 WSH - Preenchimento dos campos da tea de consulta do sistema GID 56](#_Toc392351452)

[Quadro 13 Salvar os dados em arquivo texto 56](#_Toc392351453)

[Quadro 14 Executar ação salvar no programa Bloco de Notas 57](#_Toc392351454)

[Quadro 15 *Scrit* para testar se o arquivo foi criado 57](#_Toc392351455)

[Quadro 16 Exemplo de script para abrir o programa Bloco de Notas 59](#_Toc392351456)

[Quadro 17 Uso do comando WinWait 59](#_Toc392351457)

[Quadro 18 Preenchimento dos campos da janela de pesquisa do GID 60](#_Toc392351458)

[Quadro 19 Execução do comando salavar no programa Bloco de Notas 60](#_Toc392351459)

[Quadro 20 Execução do programa AutoHotkey através da WSH 60](#_Toc392351460)

Lista de Abreviaturas e Siglas

CNAB Centro Nacional de Automação Bancária

COM Component Object Model

CRM Customer Relationship Management

EDI Electronic Data Interchange

ERP Enterprise Resource Planning

GID Gerenciamento de Informações do DETRAN

HTTP Hypertext Transfer Protocol

ODBC Open Database Connectivity

OLAP On-Line Analytical Processing

RPC Remote Procedure Call

SGBD Sistemas Gerenciadores de Banco de Dados

SOAP Simple Object Access Protocol

WSDL Web Services Description Language

WSH Windows Script Host

XML eXtensible Markup Language

VBA Visual Basic for Applications

Sumário

[Introdução 11](#_Toc392351461)

[1 INTEGRAÇÃO DE SISTEMAS 13](#_Toc392351462)

[1.1 Tipos de Integração 17](#_Toc392351463)

[1.1.1 Integração pela base de dados 18](#_Toc392351464)

[1.1.2 Troca de mensagens 20](#_Toc392351465)

[1.1.3 Transferência de arquivos 21](#_Toc392351466)

[1.1.4 Procedimento remoto por chamada 21](#_Toc392351467)

[1.1.5 Webservices 22](#_Toc392351468)

[1.1.6 WSH 24](#_Toc392351469)

[2 AMBIENTE DO PROBLEMA 25](#_Toc392351470)

[2.1 Processo realizado no sistema DETRAN 27](#_Toc392351471)

[2.2 Processo realizado no sistema CFC 29](#_Toc392351472)

[3 WINDOWS SCRIPT HOST 35](#_Toc392351473)

[3.1 Elementos XML (XML *Elements*) 37](#_Toc392351474)

[3.2 Objetos (*Objects*) 38](#_Toc392351475)

[Propriedades (*Properties*) 40](#_Toc392351476)

[Métodos (*Methods*) 40](#_Toc392351477)

[Eventos (*Events*) 40](#_Toc392351478)

[3.3 Mensagens de Erro (*Error Messages*) 41](#_Toc392351479)

[3.4 Exemplos de *Scripts* 41](#_Toc392351480)

[3.4.1 *Script* para mapear unidade da rede 41](#_Toc392351481)

[3.4.2 Criar planilha Excel 42](#_Toc392351482)

[3.4.3 Consulta em banco de dados Oracle 43](#_Toc392351483)

[3.4.4 Execução e gerenciamento 45](#_Toc392351484)

[3.5 Utilização da WSH para solucionar o problema proposto neste trabalho 46](#_Toc392351485)

[4 DESENVOLVIMENTO E TESTES 48](#_Toc392351486)

[4.1 Utilização da WSH para integrar os sistemas CFC e GID 49](#_Toc392351487)

[4.1.1 Tipos de declaração 49](#_Toc392351488)

[Declaração do tipo *Early Binding* 49](#_Toc392351489)

[Declaração do tipo Late Binding 50](#_Toc392351490)

[Diferenças entre o uso da Late Binding e Early Binding 51](#_Toc392351491)

[4.1.2 Dificuldades encontradas 51](#_Toc392351492)

[4.2 Desenvolvimento da solução proposta 55](#_Toc392351493)

[4.3 Testes no ambiente do CFC 57](#_Toc392351494)

[4.4 Resultados dos testes realizados *in loco* 61](#_Toc392351495)

[CONCLUSÃO 63](#_Toc392351496)

[Referências Bibliográficas 65](#_Toc392351497)

[apendíces 67](#_Toc392351498)

[Código fonte para gerar *Script* integração GID utilizando a WSH 67](#_Toc392351499)

[Código fonte para gerar *Script* integração GID utilizando a WSH e o AutoHotkey 69](#_Toc392351500)

[Código fonte para executar a crítica dos dados exportados do GID 71](#_Toc392351501)

[Código fonte para verificar conflitos entre os horários 75](#_Toc392351502)

[Procedimento operacional para integração entre os sistemas CFC e GID 77](#_Toc392351503)

Introdução

O DETRAN-RS, Departamento Estadual de Trânsito, é uma autarquia do Governo do Estado do Rio Grande do Sul, responsável pelo credenciamento e fiscalização dos CFC – Centro de Formação de Condutores. Os CFCs são responsáveis pelas aulas teóricas e práticas para emissão da Carteira Nacional de Habilitação - CNH. Os CFCs utilizam o sistema GID – Gerenciamento de Informações do DETRAN para o lançamento das aulas. (DETRAN, 2013a)

A estrutura organizacional de um CFC é composta de diretores, psicólogos, médicos, instrutores teóricos, instrutores práticos e atendentes. Os diretores são responsáveis pelo gerenciamento e administração do CFC. O Diretor de Ensino do CFC é responsável por zelar pelo cumprimento das normas para emissão da CNH, além da responsabilidade sobre a ação dos instrutores. Os atendentes realizam os cadastros, lançam as datas das aulas teóricas e práticas, realizam a marcação de prova dentre outras atividades. Os instrutores ministram as aulas teóricas e práticas e acessam o sistema GID para lançar as aulas realizadas, utilizando sua senha e usuário cadastrados.

A solução proposta neste trabalho será implantada no ambiente de um CFC, tendo por objetivo automatizar o processo de conciliação de lançamento de aulas entre o sistema gerencial do CFC e o sistema DETRAN (GID). Atualmente o processo é realizado de forma manual pelo Diretor de Ensino do CFC, pois não há possibilidade de integração direta entre os sistemas. A integração visa automatizar tarefas desempenhadas pelo usuário buscando maior eficiência e redução no tempo gasto pelo mesmo.

Segundo Martins (2005), as empresas buscam integrar sistemas, mesmo quando estes não foram projetados para a integração. Elas desenvolvem soluções que em muitos casos não seguem as normas técnicas, a fim de possibilitar que a informação possa ser compartilhada.

O presente trabalho automatiza o processo de auditoria através da utilização da linguagem de *script* Windows Script Host (WSH). A WSH é uma linguagem de *script* desenvolvida pela *Microsoft* voltada para a criação de *scripts* para o sistema operacional *Windows.* Por apresentar recursos que possibilitam gerenciar a interfaces de outros programas, ela foi escolhida para ser utilizada na integração entre os sistemas CFC e DETRAN.

Ao utilizar a *WSH* pretende-se reduzir o tempo gasto na realização da auditoria, tornando possível que a mesma seja feita diariamente, ao invés de semanalmente, como atualmente. Assim, permitirá ao Diretor de Ensino do CFC que se dedique a outras atividades durante a execução do processo automatizado. As principais atividades realizadas pelo Diretor de Ensino compreende a organização pedagógica dos cursos ministrados, suporte e apoio aos instrutores, fiscalização das atividades exercidas pelos instrutores, a identificação de irregularidades e sua comunicação ao DETRAN.

A importância deste trabalho ao CFC consiste em fornecer uma ferramenta que sirva para auxiliar no controle dos procedimentos de lançamento de aulas no sistema GID. Possibilitará também que sejam tomadas medidas preventivas junto ao DETRAN, evitando sanções por erros em lançamentos no sistema GID. Desta forma o CFC demonstra ao DETRAN que não é conivente com os instrutores que não seguem o que determinam as portarias emitidas pelo órgão.

Este trabalho está dividido em quatro capítulos. No capítulo 1 será apresentada a pesquisa já existente sobre integração de sistemas e o referencial teórico sobre o assunto. No capítulo seguinte será abordado o problema a ser resolvido e a caracterização do ambiente onde a implantação será realizada. O capítulo 3 apresentará a linguagem de *script* Windows Script Host, o histórico de versões, a sua estrutura e sintaxe dos comandos além de exemplos de utilização. O desenvolvimento da solução proposta será apresentado no capítulo 4 bem como os testes e validações realizados no CFC.

# INTEGRAÇÃO DE SISTEMAS

As empresas estão em constante transformação na busca por manter-se competitivas. Neste cenário, os sistemas de informações tornam-se indispensáveis. No início da era da informatização, os sistemas cumpriam apenas o papel de gerenciar os processos produtivos, emitir consultas e relatórios gerenciais. A evolução dos mercados, que cada vez mais exige que as empresas sejam competitivas e eficientes, fez com que os sistemas evoluíssem para suprir este novo panorama. Laudon e Laudon (2010, p.12) definem sistemas de informação da seguinte forma:

Um sistema de informação (SI) pode ser definido tecnicamente como um conjunto de componentes inter-relacionados que coletam (ou recuperam), processam. armazenam e distribuem informações destinadas a apoiar a tomada de decisões, a coordenação e o controle de uma organização. Além de dar apoio à tomada de decisões, à coordenação e ao controle, esses sistemas também auxiliam os gerentes e trabalhadores a analisar problemas, visualizar assuntos complexos e criar novos produtos.

Segundo Turban, Mclean e Wetherbe (2004) os sistemas de informação são criados com o objetivo de alcançar determinados objetivos. Destacam ainda que entre os principais objetivos está o da “transformação, de maneira econômica, de dados em informação e conhecimento”. Estes conceitos são definidos por Turban, Mclean e Wetherbe (2004) como:

* Dados são itens referentes a uma descrição primária de objetos, eventos, atividades e transações que são gravados, classificados e armazenados, mas não chegam a ser organizados de forma a transmitir algum significado específico. Os dados podem ser numéricos, alfanuméricos, figuras, sons ou imagens. Um banco de dados consiste de dados armazenados, organizados de maneira a possibilitar sua posterior recuperação.
* Informação é todo conjunto de dados organizados de forma a terem sentido e valor para seu destinatário. Este interpreta o significado, tira conclusões e faz deduções a partir deles. Os dados processados por um programa aplicativo têm uso mais específico e maior valor agregado do que aqueles simplesmente recuperados de um banco de dados. Esse aplicativo pode ser um sistema de gerenciamento de estoques, um sistema de matrículas *online* de urna universidade, ou um sistema de Internet para compra e venda de ações.
* Conhecimento consiste de dados e informações organizados e processados para transmitir compreensão, experiência, aprendizado acumulado e técnica, quando se aplicam a determinado problema ou atividade. Os dados processados para extrair deduções críticas e para refletir experiência e perícia anteriores fornecem a quem os recebe conhecimento organizacional, de alto valor potencial . Atualmente, a gestão do conhecimento é um dos tópicos mais valorizados na área de Sistemas de Informação.

Os sistemas de informação, de acordo com Turban, Mclean e Wetherbe (2004, p.62), podem ser compostos de várias configurações diferentes, além de utilizar uma diversidade de sistemas. O grau de complexidade pode ser determinado por fatores técnicos ou econômicos.

Os sistemas de informação, outrora voltados ao papel gerencial, passaram a ter um enfoque estratégico. Porém, é comum, devido à complexidade das empresas, que estas utilizem mais de um sistema para gerenciar as suas informações. Spackman e Speaker (2006) comentam que o uso de mais de um sistema independente é comum nas empresas devido à diversidade dos ativos de informação. Os autores citam como exemplo o uso de banco de dados, sistemas gerenciais, sistemas *WEB*, planilhas eletrônicas, dentre outros, tornando a integração entre eles uma necessidade. Turban, Mclean e Wetherbe (2004, p.276) enfatizam que, durante muitos anos, as empresas desenvolviam sistemas de forma separada a fim de atender a demandas específicas das diversas áreas da empresa.

A integração de sistemas é definida por Spackman e Speaker (2006) como “a construção de interfaces para sistemas importantes e o uso dessas interfaces para fazer transferências de dados em lote entre esses sistemas”. A importância da integração, de acordo com Spackman e Speaker (2006), está na maximização do uso dos dados, não apenas na sua reutilização, mas na garantia da qualidade da informação. A integração propicia a propagação da informação pelos diversos sistemas da empresa, além de possibilitar a racionalização na entrada de dados, uma vez que não será necessário inserir a informação em diversos sistemas.

Segundo afirmam Turban, Mclean e Wetherbe (2004, p.276), as integrações de sistemas funcionais isolados representam um grande problema para as empresas, pois existem dificuldades técnicas, problemas em integrar diferentes tipos de dados, políticas de compartilhamento da informação, identificação de quem tem prioridade no processamento e de que forma o usuário irá interagir com a integração.

A implantação de uma solução integrada deve ser planejada a fim de garantir a eficácia da solução. Souza e Saccol (2003, p.46) sugerem as seguintes etapas:

* Levantamento dos processos internos;
* Identificação do modelo necessário de dados;
* Identificação da proposta de solução dos principais problemas relacionados com o fluxo de informação;
* Proposta de indicadores de desempenho de gestão;

O processo de implementação de uma integração de sistemas pode ser dividido em três fases, como podemos observar na figura 1. Este modelo foi desenvolvido por Souza e Zwicker (2000 apud Souza e Saccol, 2003) para a implantação de sistemas de ERP, mas podemos adaptar este modelo para a integração de sistemas.

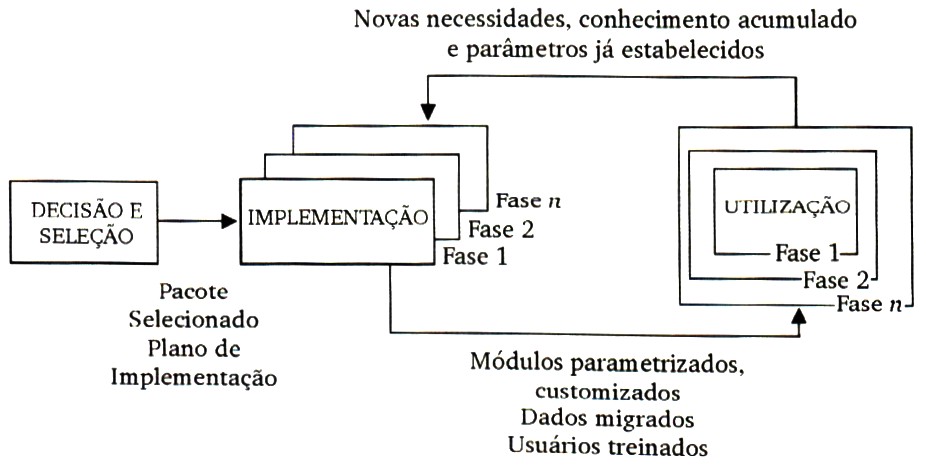


Figura 1 Fases de Implementação

Fonte: Souza e Zwicker (2000 apud Souza e Saccol, 2003)

Na fase de decisão, são definidos os critérios para integração, levantamento dos dados a serem integrados, os custos de implementação, os requisitos dos sistemas, a arquitetura técnica e a qualidade do suporte. Na fase de implementação são realizadas as configurações do hardware e software, realização de testes, treinamento e suporte ao usuário, além da documentação de todos os pontos importantes. Na fase de utilização o sistema passa a ser utilizado no dia a dia da empresa. Esta utilização poderá apresentar novas demandas e ou alterações, que, por sua vez, realimentarão a etapa de implementação.

Souza e Zwicker (2000 apud Souza e Saccol, 2003) ressaltam que neste modelo é possível realizar a implementação em fases. Neste caso, deve se considerar as limitações impostas pelas fases anteriores ao desenvolver novas fases.

A integração de sistemas apresenta consequências que devem ser levadas em conta pelas empresas. Estas consequências não representam algo negativo, mas servem como alerta para que a empresa crie mecanismos para evitá-las e/ou minimizar o seu impacto no uso da integração. (SPACKMAN e SPEAKER, 2006)

De acordo com Spackman e Speaker (2006), as consequências da integração são:

* Mais propensa a erros: Não quer dizer que ocorram mais erros em um ambiente integrado, mas sim que, quando dados errados são introduzidos, eles podem se propagar rapidamente por toda a empresa;
* Menos segura: Um ambiente integrado é menos seguro justamente porque é integrado. Como os sistemas esperam comunicar-se entre si, eles se tornam vulneráveis caso um sistema associado esteja comprometido;
* Mais frágil: Os sistemas que estão interligados terão impacto uns sobre os outros. Uma solução de integração bem feita minimizará tais efeitos;
* Utiliza mais recursos: A integração usa mais recursos de sistema. A comunicação de eventos e mensagens por uma organização integrada exige poder de computação, memória, largura de banda de rede e espaço em disco extras;
* Mais difícil de modelar e testar: Os sistemas integrados são difíceis de modelar simplesmente devido ao número de sistemas, pessoas e processos, assim como o volume de dados, envolvidos. Os sistemas integrados também podem ser difíceis de testar, pois é improvável que você tenha um laboratório de testes que imite a infra-estrutura da organização inteira.

As desvantagens e dificuldades, segundo Spackman e Speaker (2006), não devem ser um impedimento para realizar a integração, pois estes problemas podem ser resolvidos ou minimizados através de um planejamento bem elaborado. A integração traz como principais vantagens os ganhos em funcionalidades, produtividade e confiabilidade.

A integração propicia a empresa um maior controle do fluxo de dados, podendo identificar erros e corrigi-los na sua origem, propagando a sua correção nos sistemas integrados. Segundo Turban, Mclean e Wetherbe (2004, p.276)

Qualquer método utilizado para integrar sistemas de informação ajuda a reduzir custos, aumentar a produtividade dos funcionários e facilitar o compartilhamento de informação e a colaboração, tão necessários para melhorar os serviços.

As principais vantagens da integração de sistemas para as empresas são: (A INTEGRAÇÃO, 2006)

* Qualidade da informação produzida. A qualidade se dá pela obtenção de informação mais detalhada e rigorosa, integridade e coerência da informação existente nos vários ssrviços;
* Comunicação: dinamização da comunicação entre departamentos e serviços, permitindo uma organização por processos e orientando toda a empresa para a mesma estratégia de negócio;
* Acesso à informação em tempo real: acesso da gestão de topo a dados atualizados, confiáveis e imediatos sobre o estado do negócio;
* Redução de tempo, esforços e erros na execução de tarefas: sistemas integrados funcionam em conjunto permitindo um melhor desempenho da organização, evitando a duplicação de esforços na inserção de informação em sistemas diversos;
* Facilidade: a informação integrada é fácil de manter e controlar, evitando a multiplicação dos mesmos registros por diferentes bases de dados. Sistemas dispersos, que não comunicam entre si, dificultam análises mais complexas que proporcionam indicadores úteis de gestão;
* Impulso para novos negócios: a interatividade facilita a realização de novos negócios. Através do cruzamento e partilha de conhecimentos promove-se os cross selling (perceber as necessidades dos clientes) de produtos complementares e antecipam-se oportunidades de negócios.

## Tipos de Integração

A integração de sistemas pode se dar de várias formas diferentes. A escolha por qual forma utilizar, depende de questões tecnológicas dos sistemas envolvidos. As principais opções utilizadas para integração de sistemas são:

### Integração pela base de dados

A integração pela base de dados pode ser realizada por compartilhamento da base de dados. O acesso a base de dados pode ser através da utilização do ODBC (acrônimo para *Open Database Connectivity)*, JDBC (Java DataBase Connectivity) e OLAP (*On-Line Analytical Processing*).

ODBC é um padrão criado pela Microsoft para possibilitar o acesso a diferentes bases de dados, tais como ORACLE, SQL Server, PostgresSql, MySql, dentre outros. O ODBC funciona como uma camada intermediária (figura 2) entre a aplicação e o banco de dados. Isto faz com que um sistema possa acessar as informações de várias bases de dados diferentes, O padrão utilizado para acessar os dados é o SQL (*Structure Language Query*) A utilização do ODBC pressupõe a instalação de um *driver* específico para o banco de dados que se deseja acessar. A figura 2 apresenta como ocorre a comunicação entre a aplicação e a base de dados.

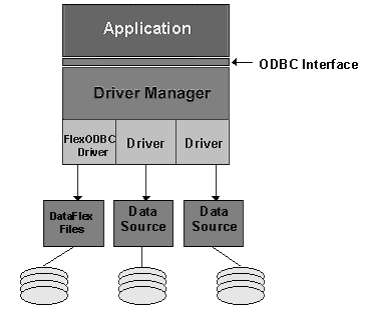


Figura 2 ODBC - Modelo Funcional

Fonte: ODBC 2.0 (Visigenic Software, 2000)

As principais vantagens do uso do ODBC, segundo a MSDN (2013) são:

* Permitir a aplicações para Windows acesso a múltiplos dados através de um método simples sem considerar os diversos formatos dos arquivos de dados;
* Superar o problema de bancos de dados diferentes que têm meios diversos de prover acesso para a informação contida neles;
* Simplificar o acesso de forma que o usuário não tenha necessidade de um alto grau de conhecimento técnico para poder ter acesso a diferentes bancos de dados;
* Udo do SQL é usado como método de acesso aos dados através do ODBC;
* Simplifica o desenvolvimento de aplicações provendo acesso fácil a múltiplas bases de dados;
* Desvincula as aplicações de mudanças de versões na base de dados. Mudanças nos serviços da rede, servidores e base de dados não impactarão aplicações usando ODBC. Os investimentos da empresa em bases de dados existentes não são perdidos;
* Os conhecimentos adquiridos pelos desenvolvedores nos bancos de dados continua sendo válidos.

A Oracle (2013) define o JDBC como sendo uma API JAVA utilizada para acessar banco de dados relacionais com suporte ao SQL. As principais funções da API são:

* Conectar a uma fonte de dados, como um banco de dados;
* Enviar consultas e instruções de atualização no banco de dados;
* Receber e processar os resultados recebidos do banco de dados em resposta a sua consulta.

A utilização do JDBC permite acessar os principais SGBD (Sistemas Gerenciadores de Banco de Dados) do mercado. Assim como ocorre com o ODBC, o banco de dados deve oferecer suporte ao JDBC. A figura 3 demonstra como a aplicação JAVA acessa o SGBD.

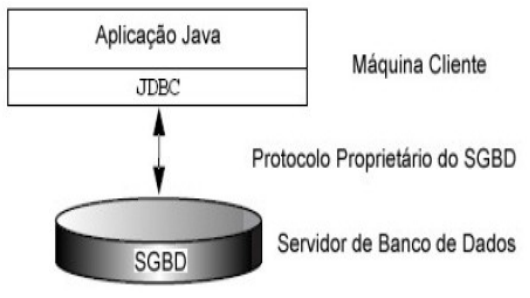


Figura 3 JDBC Modelo Funcional

Fonte: Oracle (2013)

A integração também pode ser realizada por OLAP (acrônimo para *On-Line Analytical Processing*), uma tecnologia utilizada para facilitar a consulta dinâmica e em tempo real em uma base de dados. Jennings (2000, p.25) define OLAP como sendo um cubo de dados (*DataCubes*). Os cubos de dados são visões multidimensionais de dados agregados, organizados de forma não relacional (hierárquica) utilizada para gerar informações analíticas, apurar tendências. Singh (200l) conceitua OLAP como:

"O Processamento Analítico On-Line é o próximo passo lógico além das consultas e relatórios e é o próximo estágio evolutivo em direção a uma solução completa de suporte à decisão. As ferramentas OLAP fornecem meios tecnológicos para análise complexa do negócio, possibilitando aos usuários analisar e navegar os dados para detectar tendências, exceções e obter detalhes para entender melhor os altos e baixos das atividades de seu negócio"

O OLAP é muito utilizado em sistemas ERP, CRM e *Data Warehouses*. Esta opção de integração é utilizada quando se tem um grande volume de dados, quando os sistemas envolvidos podem compartilhar a informação armazenada e necessita-se de rapidez na busca pela informação.

### Troca de mensagens

Neste modelo de integração, é utilizado um servidor de mensagens (*Middleware - Message Oriented Middleware – MOM*). Faerman (2013) descreve que na integração por troca de mensagens, a aplicação que precisa de dados ou funcionalidades de outra, envia uma mensagem. A mensagem enviada é processada pela aplicação de destino, que atende à requisição e, conforme a solicitação, envia uma mensagem de resposta. Este processo é demonstrado na figura 4.

A vantagem deste modelo de integração é o fato de as aplicações envolvidas na integração não ficarem bloqueadas em caso de falhas, permitindo aos sistemas continuarem a executar outras tarefas. Quando ocorrem falhas na comunicação, as mensagens ficam armazenadas no cliente e/ou no servidor até que o serviço seja reestabelecido. Quando o serviço é reestabelecido, as mensagens são enviadas. Um exemplo de aplicação que utiliza este tipo de integração é o serviço de E-mail. As principais características da integração por troca de mensagens são: (FAERMAN, 2013)

* entrega de mensagens a diversos destinos;
* garantia de entrega e persistência das mensagens.



Figura 4 - Integração por troca de mensagens

### Transferência de arquivos

A integração por transferência de arquivos possibilita aos sistemas heterogêneos trocar informações através da utilização de arquivos. Estes são criados conforme o *layout* definido entre as empresas. A integração por troca de arquivos mais conhecida é o EDI (*Electronic Data Interchange*). O padrão CNAB adotado pelos bancos é outro exemplo de integração por troca de arquivos. Neste tipo de integração, o sistema A gera o arquivo para ser importado no sistema B.

### Procedimento remoto por chamada

A integração por procedimento remoto de chamada (*Remote Procedure Call - RPC*) é um padrão que utiliza um servidor que disponibiliza o serviço o qual pode ser acessado por outras aplicações. A comunicação neste modelo é síncrona, ou seja o sistema envia a requisição, aguarda o retorno para enviar nova requisição ou encerrar a comunicação, conforme demonstrado na figura 5. O RPC é um modelo baseado em cliente-servidor. Neste modelo além do compartilhamento dos dados, é possível compartilhar funcionalidades completas. Faerman (2013) descreve que no RPC, os objetos são publicados para que seus métodos possam ser invocados remotamente através da rede. A invocação remota é tratada por classes intermediárias geradas automaticamente (*stubs*), acessadas pelo cliente RPC. Esta estrutura torna o código da invocação de um método remoto diferente da invocação de um método local. Um exemplo de uso de RPC são as bibliotecas RMI, EJB e JAX-RPC utilizadas pelo Java para publicação e invocação remota.

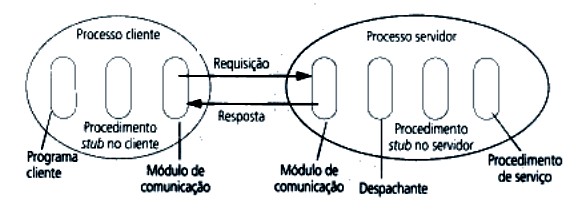


Figura 5 Exemplo de RPC

Fonte: Coulouris, Dollimore e Kindberg (2007)

### Webservices

A integração de sistemas através de Webservices é a forma que mais tem sido adotada para trocar informações entre sistemas atualmente (Neto, 2013). O Webservice possibilita a comunicação entre sistemas diferentes, que podem ser executados em plataformas diferentes. A comunicação por Webservice utiliza o padrão XML (*Extensible Markup Language* - Linguagem Extensível de Marcação). Martins (2006, p.111) define Webservice como sendo:

Um Web Service representa uma interface descritiva de um conjunto de funções acessíveis por vários canais heterogéneos. A norma XML é utilizada para a especificação do Web Service, que é invocado para executar uma determinada tarefa ou um conjunto de tarefas e obter um resultado específico.

Neto (2013, p1) enfatiza que a grande vantagem do Webservice é possibilitar a integração de sistemas distintos em plataformas heterogêneas.

Os Web Services é uma tecnologia que tem por objetivo integrar sistemas distintos através da Internet usando protocolos padronizados que garantem a independência de plataforma e de linguagem de programação em que esses sistemas foram escritos. Assim, é possível disponibilizar uma coleção de métodos em um servidor remoto e permitir que sejam acessados por programas clientes. (NETO, 2013, p.1)

Os principais componentes do Webservice são:

* SOAP (*Simple Object Access Protocol*): é o protocolo utilizado para a comunicação com o servidor. Responsável por processar as requisições e respostas entre os métodos;
* REST ([*REpresentation State Transfer*](http://en.wikipedia.org/wiki/Representational_State_Transfer)): protocolo baseado no protocolo HTTP. Utilizado para fazer a comunicação entre o cliente e o servidor.
* XML (*eX-tensilbe Markup Language*): é a linguagem padronizada para troca de dados multiplataforma.
* WSDL (*Web Services Description Language*): é uma interface padronizada que permite aos programas clientes encontrarem os métodos remotos do Webservice;

A comunicação entre o cliente e o servidor de Webservice é realizada através da internet, por intermédio do protocolo HTTP, responsável pelo transporte do arquivo XML.

Neto (2013) demonstra como funciona a integração por uso de Webservice na figura 6. As chamadas do cliente são codificadas em SOAP e transmitidas por HTTP, quando o serviço recebe a requisição acontece o inverso, as chamadas são decodificadas executadas e a resposta é codificada novamente para ser enviada para o cliente.

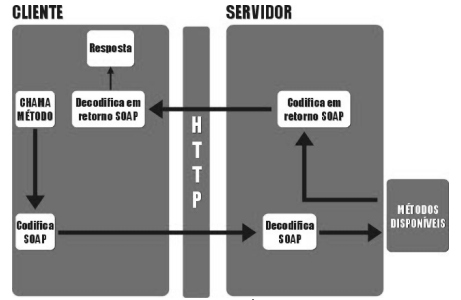


Figura 6 Funcionamento do WebService

Fonte: Neto (2013, p. 2)

Na figura 7 é demonstrado o acesso de vários Webservices por um sistema. Esta é uma das vantagens do Webservice, pois a comunicação ocorre de forma assíncrona, permitindo a um servidor responder a vários clientes. Um exemplo de uso de Webservice é a Nota Fiscal Eletrônica.

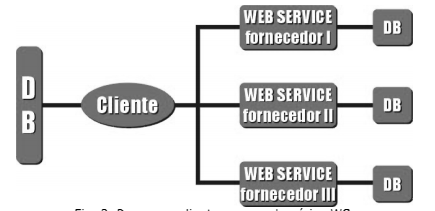


Figura 7 Programa cliente acessando vários Webservices

Fonte: Neto (2013, p2)

### WSH

A WSH é uma linguagem de *script* que oferece suporte para execução no ambiente gráfico do Windows. Foi criada para possibilitar a criação de *scripts* voltados para a automação de tarefas do sistema Operacional Windows. Porém, por apresentar muitos recursos, a linguagem pode ser utilizada para realizar tarefas como acesso a banco de dados, criar documentos em vários formatos, manipular a execução e gerenciamento de programas através de comandos que possibilitam gerenciar a *interface*. A linguagem apresenta ainda suporte ao padrão XML.

Esta linguagem pode ser executada de três formas. A primeira através da linha de comando do sistema operacional. A segunda pela execução no ambiente gráfico do sistema. A terceira forma é criar uma instância no código de programação de linguagens que tem suporte a execução do *script*. Dentre as linguagens com suporte, podemos citar o Delphi, Visual Basic, VBA e PHP.

Devido ao sistema do DETRAN não disponibilizar nenhuma forma de integração como as descritas neste capítulo, será utilizada a linguagem WSH para realizar a integração entre o sistema citado e o sistema gerencial do CFC. A linguagem WSH será utilizada para acessar o sistema do DETRAN simulando a ação do usuário. Para isto serão utilizados os recursos da WSH para acessar os menus, gerar as consultas e exportar dados como se a tarefa estivesse sendo realizada pelo próprio usuário.

# AMBIENTE DO PROBLEMA

Neste capitulo será descrito o ambiente do problema envolvendo o processo de auditoria entre os sistemas do CFC e o do DETRAN (GID). A estrutura hierárquica do CFC é composta de diretores, instrutores, atendentes, médicos e psicólogos. Os diretores são responsáveis por revisar os procedimentos, dar suporte as atendentes e aos instrutores além de fazer cumprir todas as normas e portarias do DETRAN. As atendentes fazem o atendimento ao cidadão, fanzendo a abertura do serviço, o lançamento das aulas e demais rotinas. Os instrutores são responsáveis por inistrar as aulas e fazer o lançamento no sistema GID.

O sistema CFC é o sistema gerencial específico do CFC, utilizado para gerenciar as compras, o planejamento das aulas e os recebimentos. Neste sistema as atendentes lançam a programação das aulas. O sistema GID é o sistema de informações fornecido pelo DETRAN no qual são informados o andamento dos serviços referentes a concessão de habilitação. Os instrutores lançam as aulas realizadas no sistema GID.

A fim de apurar se os instrutores realizaram as aulas previstas, torna-se necessário realizar a auditoria entre os dados lançados no sistema CFC e GID. A auditoria busca essencialmente identificar aulas planejadas e não ministradas além de outras inconsistências. Atualmente o processo de auditoria é realizado de forma manual pelo Diretor de Ensino do CFC, pois não há possibilidade de integração direta entre os sistemas.

A integração visa automatizar tarefas desempenhadas pelo Diretor de Ensino do CFC buscando maior eficiência e redução no tempo gasto pelo mesmo. É importante ressaltar que a consulta aos dados da marcação de aulas no sistema GID só podem ser realizadas pelos instrutores e pelos Diretores do CFC.

O processo atual consiste em o Diretor de Ensino entrar no sistema GID, acessar a opção de consulta por instrutor, informar o RG deste e o período das aulas realizadas. Posteriormente é realizada a cópia dos dados em tela e salva em um arquivo texto. Após realizar todas as exportações dos arquivos, o Diretor de Ensino entra no sistema gerencial do CFC para realizar a conciliação dos dados. O processo de conciliação consiste em verificar se os lançamentos das aulas práticas em ambos os sistemas conferem. O processo executado nos dois sistemas será descrito em detalhes nas seções 2.1 e 2.2.

O processo atual é demorado, pois obriga ao Diretor de Ensino realizar em média 40 consultas e exportações em cada auditoria. Esta demora faz com que a auditoria seja executada apenas uma vez por semana. O ideal seria que a mesma fosse realizada diariamente, possibilitando ao Diretor de Ensino acompanhar o trabalho dos instrutores, reduzindo sistematicamente a quantidade de erros. Outro ponto negativo é que o processo atual é suscetível a erros, pois, caso o Diretor de Ensino esqueça-se de realizar a consulta para algum dos instrutores, a conciliação não será realizada para este.

É importante ressaltar que a auditoria é fundamental, pois são os próprios instrutores que efetuam os lançamentos das aulas ministradas no sistema DETRAN. Com a auditoria o CFC faz a verificação se o que foi determinado na agenda do sistema CFC está sendo executado e se a carga horária das aulas está sendo cumprida conforme regem as normas do DETRAN. Caso a auditoria aponte para o conflito entre o horário de lançamento e o horário efetivo em que o instrutor deveria estar ministrando aulas, o CFC emite uma advertência ao instrutor e envia um comunicado ao DETRAN.

A Auditoria tem um papel importante a fim de apurar irregularidades por parte dos instrutores, identificando conflito entre os horários, observando-se a tolerância de 5 minutos. Quando ocorre a infração, o CFC deve comunicar ao DETRAN, a fim de evitar penalidades que podem ser uma advertência ou até mesmo o descredenciamento, conforme a Portaria DETRAN RS n° 70, de 13 de maio de 2002, Art. 9.º, parágrafo XVIII (DETRAN, 2013b):

Comunicar, assim que tiver conhecimento, formal e prontamente, ao Departamento Estadual de Trânsito - DETRAN-RS os fatos e as informações relevantes, caracterizadoras de desvios de conduta ou de irregularidades referentes aos processos de habilitação de condutores de veículos e demais serviços correlatos, praticadas por seus empregados, prestadores de serviço e prepostos, bem como, qualquer indício de ilícito penal ou improbidade administrativa;

Poderá o Departamento Estadual de Trânsito - DETRAN-RS, a qualquer tempo, descredenciar profissionais que demonstrem incapacidade, inabilidade ou conduta inidônea na execução de suas atividades. O CFC, ao identificar alguma irregularidade, emite advertência ao instrutor. Posteriormente encaminha cópia ao DETRAN, que tomará as providencias que poderão resultar no afastamento do instrutor conforme o Art. 12 parágrafo 1.º da portaria supracitada.

## Processo realizado no sistema DETRAN

Para realizar a auditoria, é necessário exportar os dados referentes ao lançamento das aulas do sistema GID em um arquivo texto. É gerado um arquivo para cada instrutor. Esta tarefa é realizada pelo Diretor de Ensino do CFC, pois sua senha possibilita consultar os dados de todos os instrutores.

O Diretor de Ensino acessa o sistema GID e informa seu usuário no campo “Credenciado” e senha no respectivo campo conforme a figura 8

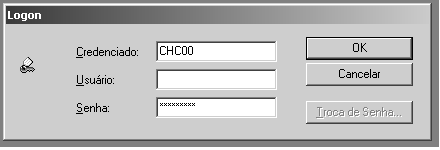
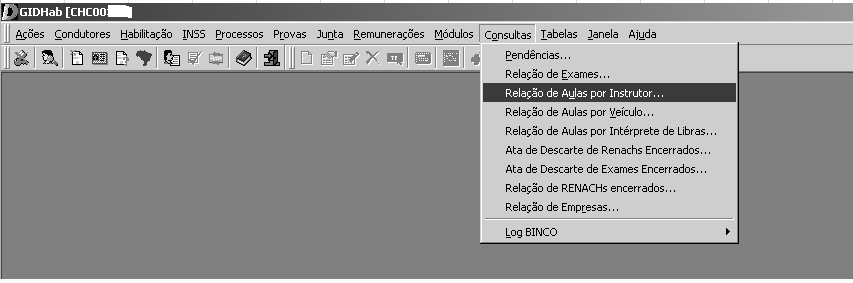


Figura 8 GID Tela de entrada do sistema

Após acessar o sistema GID, é acionado o menu Consultas, item A da figura 9, e posteriormente selecionada a opção “Relação de Aulas por Instrutor” conforme item B da figura 9.

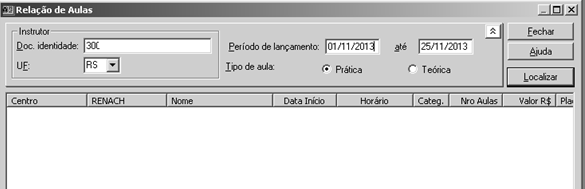


**A**

**B**

Figura 9 GID Menu para acesso a consulta

Na sequência, é apresentada a tela para realizar a consulta dos dados. Nesta tela são informados o RG do instrutor (item A da figura 10), o período de lançamento (item B da figura 10) e o tipo de aula “Prática” (item C da figura 10), para indicar que se deseja visualizar os dados referentes as aulas práticas. Ao concluir o preenchimento das informações é clicado sobre o botão localizar (item D da figura 10) para que os dados sejam visualizados.



**B**

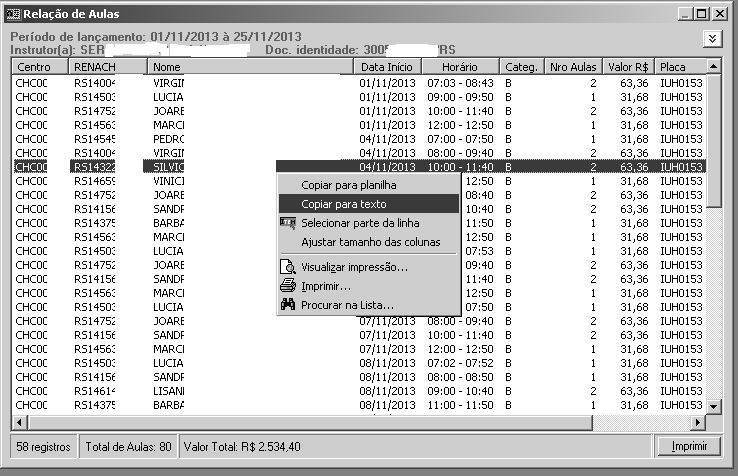
**CCDC**

**DC**

**A**

Figura 10 GID Tela para realizar a consulta

A tela da figura 11 apresenta os dados encontrados no sistema GID para os critérios determinados. Para realizar a exportação dos dados é necessário clicar com o botão direito do mouse sobre os dados (item A da figura 11) e, após, efetuada a seleção da opção “Copiar para texto” conforme item B da figura 11.



**B**

Figura 11 GID Procedimento para exportar os dados

Após executar o procedimento acima descrito, o usuário abre um editor de textos sem formatação e insere os dados. No CFC é utilizado o programa Bloco de Notas que acompanha o Windows. Para inserir os dados neste programa, pode-se utilizar o menu Editar, opção Colar ou a combinação de teclas CTRL+V. A figura 12 apresenta os dados inseridos no editor de textos.

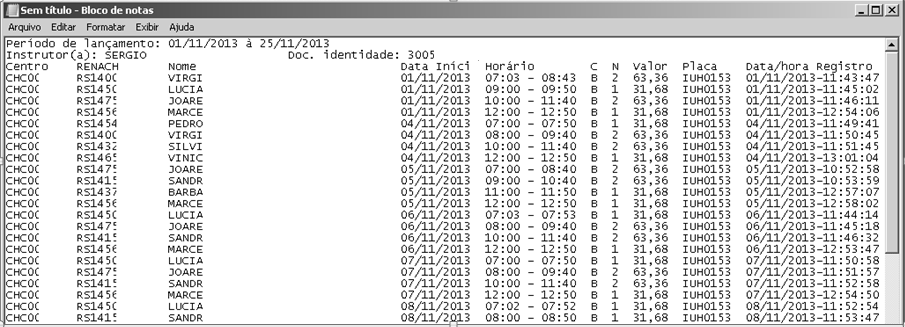
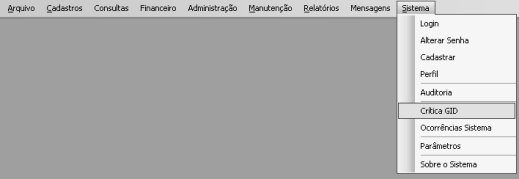


Figura 12 Dados copiados para o editor de textos

Após inserir os dados, o usuário salva o documento com o nome do instrutor em uma pasta de sua escolha. Depois de realizar este processo para cada instrutor, o Diretor de Ensino executa a crítica realizando as etapas descritas no item 2.2

## Processo realizado no sistema CFC

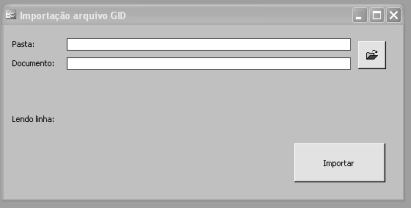
O processo de auditoria realizado no sistema CFC consiste em importar o arquivo gerado pelo sistema DETRAN, com os dados referentes aos lançamentos das aulas, de cada um dos instrutores. Após a importação é gerada a crítica, apresentando o relatório com as inconsistências. A crítica visa identificar as inconsistências entre os sistemas DETRAN (GID) e CFC. O processo consiste em armazenar em uma tabela temporária os dados importados do sistema GID. Após a importação, os dados são comparados com os dados armazenados no sistema. Desta forma, se consegue identificar lançamentos realizados em um sistema e não efetuados no outro. A importação é realizada utilizando a opção do menu “Sistema” opção “Crítica GID”, conforme demonstra a figura 13.



**A**

Figura 13 Sistema CFC - Menu Acesso Crítica GID

Ao clicar na opção “Crítica Gid”, item A da figura 13, é apresentado ao usuário a tela para selecionar o arquivo gerado pelo sistema DETRAN, contendo os dados referentes dos lançamentos das aulas, conforme a figura 14.



**A**

**B**

Figura 14 Sistema CFC - Tela para seleção do arquivo a ser importado

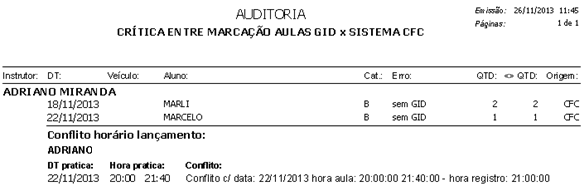
Após selecionar o arquivo, clicando no item A da figura 14, o usuário clica no botão importar (item B da figura 14) para iniciar o processamento do arquivo. O processamento armazena em uma tabela temporária as inconsistências encontradas que podem ser de três tipos:

- Lançamentos realizados no sistema CFC e não encontrados no sistema DETRAN: A ocorrência deste erro é causada quando se efetua o lançamento no sistema do CFC e não é lançado no sistema GID. A identificação deste erro é de fundamental importância para o CFC, pois de acordo com as normas do DETRAN, todas as aulas ministradas devem ser lançadas no sistema GID. O não lançamento no sistema GID pode configurar como fraude e sonegação junto ao DETRAN, uma vez que as taxas pelas aulas ministradas e não lançadas não são repassadas ao mesmo.

- Lançamentos realizados no sistema DETRAN e não encontrados no sistema CFC: Este erro identifica os lançamentos efetuados no sistema GID e não encontrados no sistema do CFC. A identificação deste tipo de erro é muito importante, pois pode representar uma aula prática ministrada e não cobrada do aluno, gerando prejuízo financeiro ao CFC.

- Lançamentos com conflito de horário: Considera-se conflito de horário os lançamentos cuja hora de lançamento no sistema DETRAN coincidem com o horário em que o instrutor deveria estar conduzindo sua aula. Este tipo de ocorrência pode indicar que o instrutor esteja lançando aula(s) no sistema que não esteja ministrando. Esta prática é totalmente proibida pelo DETRAN e, caso seja comprovada, o instrutor perde sua credencial, ficando impossibilitado de ministrar aulas.

Ao término do processamento, é apresentado ao usuário o relatório contendo somente as inconsistências encontradas como demonstrado na figura 15. No relatório são exibidos os tipos de erros acima descritos.



**C**

**B**

**A**

Figura 15 Sistema CFC - Relatório de Crítica de dados não lançados

O relatório de crítica apresenta o nome do instrutor, as datas em que ocorreram as inconsistências, a categoria, o campo “erro” (item A da figura 15), a quantidade de erros e a origem (item B da figura 15). O campo “erro” apresenta em qual sistema o mesmo ocorreu. Já o campo “origem” identifica qual sistema tem o registro lançado. Utilizando o exemplo da figura 15, podemos observar que o erro ocorreu porque os dados foram lançados no sistema CFC (campo “origem”) e não foram lançados no sistema GID (campo “erro”).

O item C da figura 15 demonstra um lançamento com conflito de horário. Esta situação ocorre quando o instrutor efetua o lançamento das aulas ministradas no sistema GID em horário que deveria estar conduzindo uma aula. Este tipo de ocorrência pode resultar em punição ao CFC caso o conflito seja maior que a tolerância permitida de 5 minutos.

A solução a ser proposta visa automatizar o processo de auditoria. Tal automatização será feita através da utilização do *script* desenvolvido com a linguagem WSH, em conjunto com o sistema CFC. Ela visa buscar os dados de todos os instrutores em período determinado pelo diretor de ensino. O processamento da crítica ocorrerá de uma única vez, sendo gerado um relatório com todas as inconsistências encontradas. Para possibilitar a automatização, o desenvolvimento será dividido em três partes como descrito abaixo.

**A primeira etapa** do desenvolvimento compreende a alteração da tela de captura do sistema CFC, onde serão incluídos os campos data inicial e data final. Esta alteração é necessária, pois as datas e o RG do instrutor serão passados como parâmetros para a execução do *script*. Os campos para seleção do arquivo serão excluídos, pois não serão mais utilizados com a automatização. O protótipo da nova tela para geração da crítica é demonstrado na figura 16.

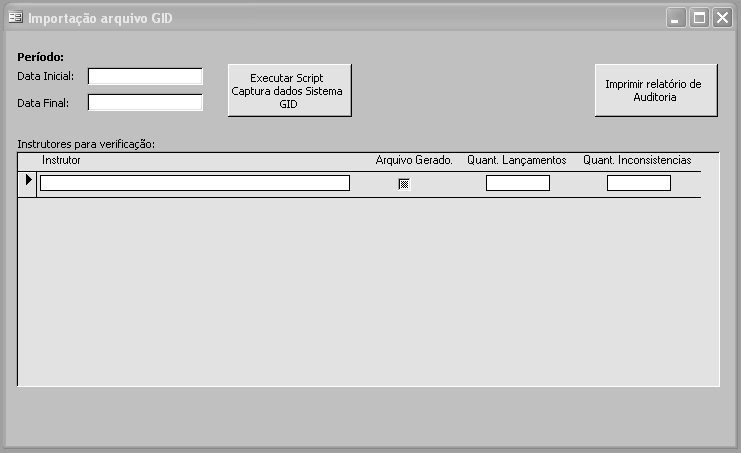


Figura 16 Protótipo nova tela de crítica entre os sistemas

**Na etapa seguinte**, será desenvolvido o *script* para captura dos dados do sistema DETRAN utilizando a linguagem de *script* WSH. O *script* será executado utilizando CScript[[1]](#footnote-1), pois o mesmo permite ser executado por linha de comando com a passagem e recebimento de parâmetros. O *script* será encarregado de buscar os dados no sistema GID e salvar os mesmos em um arquivo texto para cada instrutor. O nome do arquivo será o RG do instrutor.

**Na terceira e última etapa** será realizada a alteração da rotina de importação do arquivo texto. Nesta rotina será criada uma consulta *SQL* para buscar os dados dos instrutores ativos no cadastro do sistema. A partir do *SQL*, será criado um laço de repetição onde para cada instrutor será executado o *script* WSH com o parâmetro “RG” do instrutor e os campos data inicial e final, informados na tela. Ao clicar no botão “Executar Script Captura dados Sistema GID”, o usuário irá acompanhar a execução do *script* no quadro “Instrutores para Verificação”. Durante a execução, será apresentado o instrutor consultado (campo Instrutor), o campo para indicar se o arquivo foi gerado com sucesso (campo Arquivo Gerado), a quantidade de aulas lançadas no período (Quant Lançamentos) e a quantidade de inconsistências encontradas (campo Quant Inconsistências). Nesta etapa serão realizados os testes de execução e validação.

A integração tem como objetivo principal reduzir o tempo gasto no processo de auditoria e possibilitar que a mesma seja executada diariamente. Permitir ao diretor de ensino executar outras tarefas durante a execução da automatização. Buscar maior confiabilidade no processo, uma vez que os dados serão buscados de forma automática, além da rapidez na execução, pois a critica será executada de uma só vez, gerando um relatório único com todas as inconsistências.

A figura 17 apresenta o fluxograma de funcionamento operacional da implantação. A execução do *script* será iniciada a partir do sistema CFC.



Figura 17 Fluxograma de funcionamento *Script*

Neste capitulo foi apresentado o problema enfrentado pelo CFC para conciliar os dados entre seu sistema interno e o sistema do DETRAN. Foi abordada a importância da realização da auditoria para identificar erros no lançamento das aulas e/ou a não realização das mesmas. O processo de integração visa reduzir o tempo gasto para realizar a conciliação tornando possível a sua realização diariamente.

# WINDOWS SCRIPT HOST

A linguagem de *script* *Windows Script Host*, conhecida pela sigla WSH, é uma linguagem desenvolvida pela empresa Microsoft para o ambiente operacional Windows. Esta linguagem foi desenvolvida para suceder a linguagem de *batch* do ambiente operacional MS‑DOS e está disponível em todas as versões do sistema operacional Windows.

Segundo Veiga (2000) a linguagem foi criada com o propósito de atender administradores de sistemas, programadores e consultores na tarefa de criação de *scripts* para a automação de tarefas no sistema operacional Windows. A principal característica do WSH é o fato de ser independente de linguagem de programação. Isto significa que qualquer linguagem, desde que disponha de um mecanismo de *scripting* (interpretador), compatível com a tecnologia ActiveX, pode ser utilizada no desenvolvimento de *scripts* do WSH (Veiga, 2000).

De acordo com Veiga (2000) a WSH oferece suporte as linguagens de *script* *VBScript* e *JavaScript*, além das linguagens de programação ASP, Perl, TCL, Python, dentre outras. A 1ª versão da linguagem WSH foi lançada junto com o Microsoft Windows 98. Desde então foram liberadas várias versões da linguagem, sempre acompanhando a evolução do sistema operacional Windows. A última versão disponibilizada foi 5.8, lançada junto com o Windows 7 e mantida na versão 8 do mesmo sistema. A evolução da linguagem pode ser observada na figura 18.

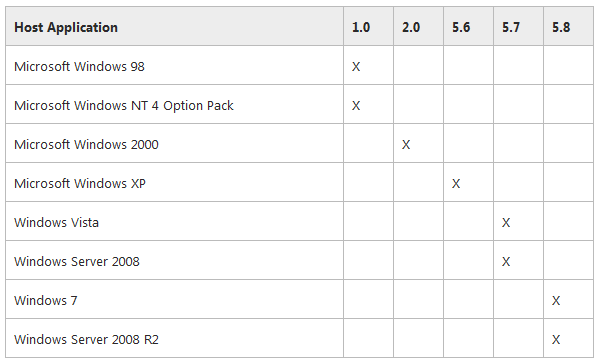
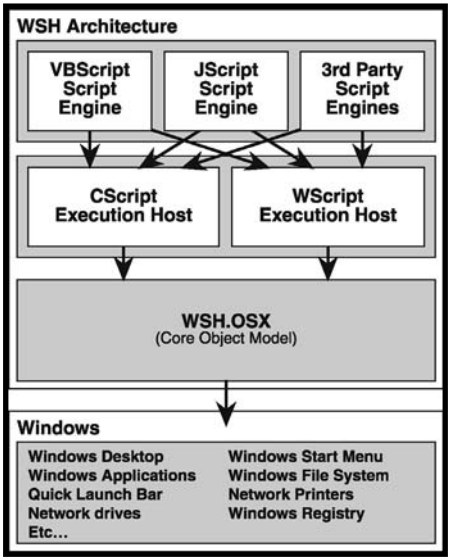


Figura 18 - Histórico de Versões do Windows Script Host

Fonte: Site da Microsoft MSDN em (MSDN, 2009a)

De acordo com a figura 19, pode-se observar que a arquitetura da WSH é dividida em mecanismo de *scripts*, *hosts* e modelo de objetos (FORD, 2008).



**C**

**B**

**A**

Figura 19 Arquitetura WSH

Fonte: Windows Script Host (JR, 2009)

O mecanismo de *scripts* é o padrão utilizado para criar o *script*. Os padrões suportados são o *VBScript*, *JScript*, *PERL* e outros, demonstrado na figura 19 item A. O *VBScript* é uma variação da linguagem de programação Visual Basic enquanto que o JScript é uma linguagem de programação similar ao Java. Segundo Battisti (2005), ao executar um *script*, o mecanismo de execução do WSH identifica o tipo de linguagem que foi utilizada e seleciona automaticamente o interpretador correto para sua execução (item C da figura 19). O *VBScript* e *JScript* são as linguagens mais utilizadas na criação de *script* com a linguagem de script WSH.

Ford (2008) define *Host* como sendo o ambiente responsável pela execução do *script*. A WSH oferece duas versões para execução dos *scripts* (figura 19 item B). A versão baseada no sistema operacional Windows utiliza o programa *WScript* enquanto que o programa *CScript* é voltado para a execução em linha de comando, compatível com o padrão do sistema operacional MS‑DOS.

Segundo Battisti (2005) a principal diferença entre as duas opções de utilização está entre a interação e a desempenho. O uso do *WScript* é mais recomendado para a execução de *scripts* que terão interação com o usuário. Já o uso do *CScript* é recomendado quando o *script* não tem interação com o usuário, pois o mesmo é executado utilizando a linha de comando do sistema operacional. Esta opção de execução permite a utilização de parâmetros para a sua execução além de apresentar maior rapidez na execução do *script*. Os parâmetros disponíveis são demonstrados na figura 20.

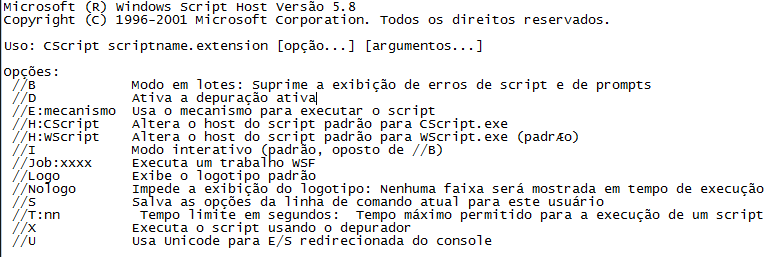


Figura 20 Parâmetros de linha de comando do *CScript*

Fonte: Sistema operacional Windows 8 - Ajuda do *CScritp*

O programa *WScript* é utilizado automaticamente pelo sistema operacional Windows. Alguns casos onde isto ocorre: clicar duplamente sobre o nome do arquivo ou utilizar a opção executar no menu iniciar do sistema supracitado.

O modelo de objetos (*Core Object Model*) possibilita o acesso aos recursos do sistema operacional Windows, pois este é composto de uma coleção de objetos. Cada objeto pode ser acessado para visualização, manipulação ou alteração (FORD, 2008). Os *scripts* podem ser criados utilizando qualquer editor de textos que permita salvar arquivos sem formatação. O arquivo contendo o *script* pode ser salvo com a extensão vbs ou js. Segundo a MSDN (2009b) a linguagem é estruturada da seguinte forma: Elementos XML (*XML Elements*), Objetos (*Objects*), Propriedades (*Properties*), Métodos (*Methods*), Eventos (*Events*) e Mensagens de Erro (*Error Messages*).

## Elementos XML (XML *Elements*)

Os elementos XML (*eXtensible Markup Language*) foram incorporados a partir da versão 5.1. Nesta versão foi incluído o analisador para XML. A vantagem de suporte ao XML está na sua flexibilidade, pois permite ao desenvolvedor criar seus próprios padrões, além de contar com suporte da maioria dos programas.

Os *scripts* criados com XML devem ser salvos com a extensão wsf. A vantagem de se criar arquivos no formato XML é que eles também podem conter códigos *VBScript* e *JScript*. Os elementos da linguagem XML são apresentados na figura 21. (VEIGA, 2000).

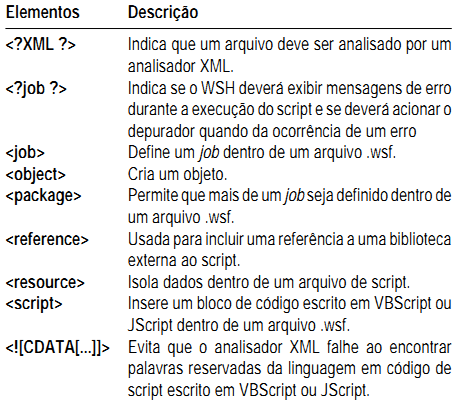


Figura 21 Elementos XML que podem ser utilizados em um *script* WSH.

Fonte: Windows Script Host (VEIGA, 2000)

## Objetos (*Objects*)

A hierarquia de objetos do WSH - *Object Model* é composta de 14 objetos conforme a figura 22. Os objetos podem ser divididos em dois grandes grupos: execução de *scripts* e funções auxiliares. O grupo de execução de *scripts* é responsável pela manipulação básica do WSH, mensagens de saída e funções básicas COM[[2]](#footnote-2). O grupo de funções auxiliares é composto de propriedades e métodos para a realização de ações que envolvem o ambiente do sistema operacional como o mapeamento de unidades de rede, alteração das chaves do registro, realização da conexão com impressoras (MSDN, 2009c).

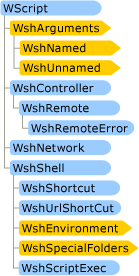


Figura 22 - Modelo de objeto do *Windows Script Host*

Fonte: Site da Microsoft MSDN (MSDN, 2009c)

A tabela 1 apresenta as características de cada objeto. (MSDN, 2009d)

Tabela 1 Objetos WSH e tarefas associadas

|  |  |
| --- | --- |
| **Objeto** | **O que você pode fazer com este objeto** |
| Wscript | * Definir e recuperar os argumentos de linha de comando * Determinar o nome do arquivo de script * Determinar o nome do arquivo host (wscript.exe ou cscript.exe) * Determinar as informações de versão de acolhimento * Criar, conectar e desconectar de objetos * Coletar eventos * Parar a execução de um script de programação * Informar a saída para o dispositivo de saída padrão (por exemplo, uma caixa de diálogo ou linha de comando) |
| [WshArguments](http://msdn.microsoft.com/en-us/library/ss1ysb2a.aspx) | Acessar todo o conjunto de argumentos de linha de comando |
| [WshNamed](http://msdn.microsoft.com/en-us/library/d6y04sbb.aspx) | Acessar o conjunto de argumentos de linha de comando chamado |
| [WshUnnamed](http://msdn.microsoft.com/en-us/library/ah2hawwc.aspx) | Acessar o conjunto de argumentos de linha de comando sem nome |
| [WshNetwork](http://msdn.microsoft.com/en-us/library/s6wt333f.aspx) | * Ligar e desligar a partir de compartilhamentos de rede e impressoras de rede * Mapa e compartilhamentos de rede mapeamento * Acessar informações sobre o usuário conectado atualmente |
| [WshController](http://msdn.microsoft.com/en-us/library/xk7bxb0d.aspx) | Criar um processo de script remoto usando o método do controlador **CreateScript** () |
| [WshRemote](http://msdn.microsoft.com/en-us/library/x9t3ze5y.aspx) | * Administrar remotamente os sistemas de computador em uma rede de computadores * Manipular outros programas / scripts |
| [WshRemoteError](http://msdn.microsoft.com/en-us/library/d02b3e15.aspx) | Acessar as informações de erro disponível quando um script remoto (um objeto WshRemote) termina, como resultado de um erro de script |
| [WshShell](http://msdn.microsoft.com/en-us/library/aew9yb99.aspx) | * Executar um programa localmente * Manipular o conteúdo do registro * Criar um atalho * Acessar uma pasta do sistema * Manipular as variáveis ​​de ambiente (como WINDIR, PATH, ou Prompt) |
| [WshShortcut](http://msdn.microsoft.com/en-us/library/xk6kst2k.aspx) | Criar um atalho |
| [WshSpecialFolders](http://msdn.microsoft.com/en-us/library/9x9e7edx.aspx) | Acessar qualquer uma das pastas especiais do Windows |
| [WshUrlShortcut](http://msdn.microsoft.com/en-us/library/265a4017.aspx) | Criar um atalho para um recurso da Internet |
| [WshEnvironment](http://msdn.microsoft.com/en-us/library/6s7w15a0.aspx) | Acessar qualquer uma das variáveis ​​de ambiente (como WINDIR, PATH, ou Prompt) |
| [WshScriptExec](http://msdn.microsoft.com/en-us/library/2f38xsxe.aspx) | Determinar o status e informações de erro sobre a execução do script com exec ()  Acessar o stdin, stdout e stderr canais |

Fonte: Site da Microsoft MSDN (MSDN, 2009c)

## Propriedades (*Properties*)

As propriedades apresentam as características físicas e formas de comportamento de um objeto.

## Métodos (*Methods*)

Os métodos são funções ou procedimentos que permitem realizar ações sobre os objetos. Estas ações podem ser realizadas para manipular os objetos, alterando suas propriedades e/ou seus comportamentos. Born (2000) destaca que o uso de métodos facilita o desenvolvimento, pois o programador não precisa saber como o objeto implementa o método. Um método pode ser aplicado a um ou mais objetos.

## Eventos (*Events*)

Os eventos representam as ações que ocorrem sobre um objeto. Estas ações podem ser interceptadas, manipuladas e tratadas através de funções e procedimentos.

## Mensagens de Erro (*Error Messages*)

O objeto *Error Messages* é responsável por retornar os erros ocorridos em tempo de execução (*run-time*). A classe oferece vários métodos para tratativas de erros. Os métodos *Raise* e *Clear* são utilizados para desativar os erros em tempo de execução.

## Exemplos de *Scripts*

A linguagem de *script* WSH foi criada com o propósito de possibilitar a automatização de rotinas e processos no sistema operacional Windows. Através dos exemplos abaixo, pretende-se mostrar que é possível utilizar a referida linguagem para executar ações além do sistema operacional, possibilitando criar *scripts* complexos para acesso a banco de dados, executar e realizar tarefas em sistemas diversos, além de criar documentos textos, planilhas entre outros.

### *Script* para mapear unidade da rede

O exemplo abaixo demonstra apresenta um *script* para realizar o mapeamento de uma unidade da rede no micro onde o mesmo é executado. Este exemplo demonstra como utilizar o WSH para realizar e automatizar tarefas no sistema operacional Windows. É possível realizar operações de configuração do ambiente operacional, do *Active Directory* e ações sobre pastas e documentos. (FORD, 2008)

‘\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

‘Script Name: DriveMapper.vbs

‘Author: Jerry Ford

‘Created: 12/07/02

‘Description: This script demonstrates how to add logic to VBScripts in

‘order to support network drive mapping.

‘\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

‘Initialization Section

Option Explicit

On Error Resume Next

Dim objWshNet

‘Instantiate the objWshNetwork object

Set objWshNet = WScript.CreateObject(“WScript.Network”)

‘Main Processing Section

‘Call the procedure that maps drive connections passing it an available

‘drive letter and the UNC pathname of the drive

MapNetworkDrive “z:”, “\\ICS\_Server\D”

WScript.Quit() ‘Terminate script execution

‘Procedure Section

‘This subroutine creates network drive mappings

Sub MapNetworkDrive(DriveLetter, NetworkPath)

‘use the objWshNetwork object’s MapNetworkDrive() method to map to drive

objWshNet.MapNetworkDrive DriveLetter, NetworkPath

End Sub

### Criar planilha Excel

Ford (2008) apresenta um outro *script* para criar uma planilha no padrão Microsoft Excel. Esta solução demonstra que qualquer programa pode criar planilhas através da utilização da linguagem WSH.

Quadro 1 WSH – Exemplo de *Script*  para criar arquivo Excel

|  |
| --- |
| ‘\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ‘Script Name: ExcelObjectModelDemo.vbs  ‘Author: Jerry Ford  ‘Created: 11/13/04  ‘Description: This script demonstrates how to use VBScript to interact  ‘ with the Microsoft Excel object model.  ‘\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ‘Initialization Section  Option Explicit  On Error Resume Next  Dim objExcel ‘Used to establish a reference to Excel Application object  Set objExcel = WScript.CreateObject(“Excel.Application”)’Instantiate Excel  ‘Main Processing Section  CreateAndHideNewSheet()  WriteExcelData()  SaveExcelSheet()  CloseSheetAndEndExcel()  TerminateScript()  ‘Procedure Section  Function CreateAndHideNewSheet()  ‘Visible is a property of the Application object. It cab be used to  ‘prevent Excel from appearing as the script executes  objExcel.Visible = False  ‘WorkBooks is a collection. Add() is a method belonging to the  ‘WorkBooks collection that opens a new empty spreadsheet  objExcel.WorkBooks.Add  End Function  Function WriteExcelData()  ‘Use the Columns object’s ColumnWidth property to set column widths  objExcel.Columns(1).ColumnWidth = 15  objExcel.Columns(2).ColumnWidth = 35  objExcel.Columns(3).ColumnWidth = 6  ‘Use the Range object’s Select method to select a range of cells  objExcel.Range(“A1:C1”).Select()  ‘Set the Font object’s Bold property  objExcel.Selection.Font.Bold = True  ‘Use the Cells object’s Value property to enter text into the  ‘spreadsheet  objExcel.Cells(1,1).Value = “Name”  objExcel.Cells(1,2).Value = “Description”  objExcel.Cells(1,3).Value = “Rating”  ‘Enter additional text  objExcel.Cells(2,1).Value = “Hangman”  objExcel.Cells(2,2).Value = “A word guessing game”  objExcel.Cells(2,3).Value = “5”  ‘Enter additional text  objExcel.Cells(3,1).Value = “TicTacToe”  objExcel.Cells(3,2).Value = “Two player board game”  objExcel.Cells(3,3).Value = “5”  ‘Enter additional text  objExcel.Cells(4,1).Value = “Blackjack”  objExcel.Cells(4,2).Value = “Player versus computer card came”  objExcel.Cells(4,3).Value = “5”  End Function  Function SaveExcelSheet()  ‘Use the ActiveWorkBook property to reference the current WorkBook.  ‘Use the WorkBook object’s SaveAs() method to save the WorkBook.  objExcel.ActiveWorkBook.SaveAs(“C:\Temp\ExcelFile.xls”)  End Function  Function CloseSheetAndEndExcel()  ‘Use the WorkBook object’s Close() method to close the spreadsheet  objExcel.ActiveWorkBook.Close()  objExcel.Quit() ‘Terminate Excel  End Function  Function TerminateScript()  WScript.Quit() ‘Terminate script execution  End Function |

### Consulta em banco de dados Oracle

Payne (2009) demonstra que a WSH pode ser utilizada além do desenvolvimento de *scripts* voltados para o sistema operacional. No exemplo abaixo, a WSH é utilizada para realizar uma consulta no banco de dados Oracle, retornando se a consulta foi realizada com sucesso.

Quadro 2 WSH Exemplo de Consulta no baco de dados Oracle

|  |
| --- |
| '#####================================================================================  '## Title: QueryOracleDatabase.vbs  '## Author: Eric Payne (Berbee Information Networks)  '## Date: 3/6/2007  '##  '## Purpose: Querys Oracle database to verify records can be returned (Poor Mans monitoring).  '## If records are returned from a simple SQL query then the database is considered'##         up and running.  '##  '## Prerequisites:  '##         1. Oracle System DSN to Oracle database  '##         2. Permission to query Oracle database  '##  '## Requirements:  '##         1. Parameter: /Query: A SQL Query to perform on database to return records  '##         2. Parameter: /DSN: A Oracle System DSN to database  '##         3. Parameter: /UserID: UserID for DSN  '##         4. Parameter: /Password: Password for DSN  '##  '## Execution:  '##     cscript //nologo QueryOracleDatabase.vbs /Query:"Select \* from Table\_Name;"  '##    /DSN:Oracle\_DSN /UserID:XXXX /Password:XXXX  '##  '## Basic Logic:  '##         1. Read in Parameters  '##         2. Attempts to open strDSN  '##         3. Attempts to run strQuery  '##         4. Outputs results  '#####============================================================================    Option Explicit  On Error Resume Next    'Arguments    Dim objArgs: Set objArgs = WScript.Arguments  if objArgs.Named.Exists("Query") Then        Dim strSQLQuery: strSQLQuery = objARgs.Named("Query")  Else        WScript.Echo "Missing /Query parameter"        WScript.Quit  End if    if objArgs.Named.Exists("DSN") Then        Dim strDSN: strDSN = objArgs.Named("DSN")  Else        WScript.Echo "Missing /DSN parameter"        WScript.Quit  End if    if objArgs.Named.Exists("UserID") Then        Dim strUserID: strUserID = objArgs.Named("UserID")  Else        WScript.Echo "Missing /UserID parameter"        WScript.Quit  End if    if objArgs.Named.Exists("Password") Then        Dim strPassword: strPassword = objArgs.Named("Password")  Else        WScript.Echo "Missing /password parameter"        WScript.Quit  End if    'Query    WScript.Echo vbnewline & "QUERYING ORACLE DATABASE"  if QueryOracleDatabase(strSQLQuery,strDSN, strUserID, strPassword) Then        WScript.Echo "  - " & strSQLQuery & " returned records"  Else        Wscript.Echo "  - " & strSQLQuery & " returned 0 records"  End if      Function QueryOracleDatabase(strSQLQuery, strDSN, strUserID, strPassword)        'Desc: Query database to see if it responds        On Error Resume Next          Dim ADODBConnection: Set ADODBConnection = CreateObject("ADODB.Connection")        Const adUseClient = 3        ADODBConnection.CursorLocation = adUseClient        ADODBConnection.ConnectionTimeout = 300          Dim strConnection: strConnection = "Provider=OraOLEDB.Oracle;Data Source=" & strDSN & \_            ";User ID=" & strUserID & ";Password=" & strPassword & ";"          ADODBConnection.Open strConnection        if err <> 0 then              WScript.Echo "An error occurred trying to open Oracle System DSN: " & strDSN & " " & \_                  err.number & " " & err.description & " " & err.Source              WScript.Quit        End if            Dim ADODBRecordSet: Set ADODBRecordSet = ADODBConnection.Execute(strSQLQuery)        if err <> 0 then              WScript.Echo "An error occurred trying to execute query: " & strSQLQuery & \_                  " on Oracle System DSN: " & strDSN & " " & err.number & " " & err.description & " " & err.Source              WScript.Quit        End if            Dim bolRecordsReturned: bolRecordsReturned = vbFalse        if ADODBRecordSet.RecordCount > 0 Then              QueryOracleDatabase = vbTrue        Else              QueryOracleDatabase = vbFalse        End if          ADODBConnection.Close        Set ADODBRecordSet = Nothing        Set ADODBConnection = Nothing    End Function |

### Execução e gerenciamento

Neste exemplo Born (2000) demonstra o uso da linguagem para manipular os programas da calculadora e do Bloco de Notas do Windows. O *script* abre os dois programas, efetua um calculo na calculadora e copia o resultado no bloco de notas.

Quadro 3 WSH Exemplo de integração entre os programas Excel e Bloco de Notas

|  |
| --- |
| // \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*  // File: SendKeys1.js (WSH 2 amostra em JScript)  // Autor: (c) G. Nascido  //  // Usando o método SendKeys eo AppActivate  // Método para lançar calculadora e bloco de notas e  // Escreve a janelas de aplicativos  // \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*  // Define as cordas título das janelas do aplicativo.  // Importante: Cordas dependem da versão localizada do Windows.  var edit\_title = "Sem título - Bloco de notas" //, / title / Window  var Calc\_Title = "Calculadora" //, título / / Window  // Cria o objeto WshShell, que Executar e  // AppActivate necessita.  var oWSH = WScript.CreateObject ("WScript.Shell");  // Tenta lançar duas aplicações. Para garantir que o  // Última aplicação recebe o foco, atrasar o script.  oWSH.Run ("notepad.exe", 1); // Lançamento Bloco de notas com janela vazia.  WScript.Sleep (500); // Atraso até Notepad recebe o foco.  oWSH.Run ("calc.exe", 1); // Lançamento Calculator.  WScript.Sleep (500); // Atraso até calculadora recebe o foco.  oWSH.SendKeys ("10"); // Apenas calcular alguma coisa.  oWSH.SendKeys ("{+}");  oWSH.SendKeys ("2");  oWSH.SendKeys ("{=}");  oWSH.SendKeys ("^{c}"); // Copiar resultado para a área de transferência.  WScript.Echo ("Escreva o texto para a janela Bloco de Notas.");  // Definir o foco para a janela Bloco de Notas.  oWSH.AppActivate (edit\_title);  // Escreve o texto a janela do Notepad.  // Primeiro inserção resultado da área de transferência.  oWSH.SendKeys ("resultado de um cálculo: ");  oWSH.SendKeys ("^{v}");  oWSH.SendKeys ("\n\n");  // Agora escreva o texto 30 vezes.  for (var i = 1; i <= 30; i++)  {  oWSH.SendKeys (i + " Olá, mundo \n");  }  WScript.Echo ("Fechar a janela da Calculadora.");  oWSH.AppActivate (Calc\_Title); // definir o foco para Calculadora.  oWSH.SendKeys ("%{F4}"); // Fechar Calculadora usando Alt + F4.  WScript.Echo ("Fechar a janela Bloco de Notas.");  // Define o foco para o bloco de notas e enviar o comando terminar.  oWSH.AppActivate (edit\_title);  oWSH.SendKeys ("%{F4}"); // Fechar o Bloco de notas usando Alt + F4.  oWSH.SendKeys ("%{S}"); // "Clique" no botão não.  oWSH.SendKeys ("Teste");  oWSH.SendKeys ("%{l}");    // End \*\*\* |

## Utilização da WSH para solucionar o problema proposto neste trabalho

A linguagem WSH, versão 5.8, utilizada para criar o *script* que irá conectar o sistema GID, executa a consulta dos dados e realiza a exportação dos mesmos em arquivo texto. O desenvolvimento do *script* busca simular a ação realizada pelo usuário no sistema GID. O diretor de Ensino do CFC acessa o sistema do CFC, seleciona o menu ferramentas, seleciona a opção “Auditoria Sistema GID”. É apresentado o formulário com os campos data inicial, data final e o botão “Executar Auditoria”.

O *script* é executado quando o usuário clica no botão. A rotina de execução realizará a conexão com o sistema GID, através de comandos que manipulam a interface do sistema, simulando a ação do usuário. É realizada uma consulta no sistema CFC para identificar os instrutores que serão consultados. Em seguida, é efetuado um laço de repetição enquanto houver instrutor a ser consultado. Para cada instrutor, é efetuada uma consulta no sistema GID, informando o RG do instrutor e o período a ser consultado. Posteriormente é realizada a exportação dos dados em um arquivo texto, nomeado com o RG do instrutor em uma pasta específica. É realizada a verificação se o arquivo existe na pasta. Caso o arquivo não exista, é apresentada a mensagem ao usuário informando que ocorreu um erro, solicitando se o mesmo deseja executar novamente a exportação. O erro pode ser causado por falha na execução do *script* pelos seguintes motivos:

- Travamento do sistema DETRAN. Ao ocorrer este erro, o sistema DETRAN deverá ser reiniciado.

- Execução do *script* em velocidade maior que o tempo necessário para exportar o arquivo com os dados do instrutor. Será necessário aumentar o tempo entre o intervalo da execução no *script*.

- Alteração da *interface* do sistema DETRAN. Neste caso, o *script* terá quer ser adaptado aos novos comandos da interface do sistema DETRAN.

Após a exportação de todos os instrutores é encerrado o sistema GID. Na sequência, é realizada a rotina de auditoria lendo os arquivos recuperados. Ao final, é apresentado o relatório da auditoria com as inconsistências encontradas.

Neste capítulo foi apresentada a linguagem de *script* WSH. Foram apresentadas a estrutura da linguagem e seus recursos. Através dos exemplos demonstrados, foi possível perceber a versatilidade da linguagem, que pode ser utilizada para criar rotinas de automatização do sistema operacional Windows, desenvolvimento de rotinas para acessar dados e criar arquivos, além de outras finalidades.

# DESENVOLVIMENTO E TESTES

A linguagem de *script* WSH oferece duas formas de execução, através do uso do *CScript*, executado a partir da linha de comando do sistema operacional *Windows*, ou através da interface gráfica utilizando o *WScript*. Para a solução proposta, será utilizada a segunda opção de execução por possibilitar a integração com outras linguagens de programação, além de permitir a realização de interações com o usuário.

A utilização do *WSCript* pode ser executada de forma direta, com a execução do arquivo do *script* ou através de outra linguagem de programação, por meio de uso de sua biblioteca. A elaboração do *script* foi desenvolvida no sistema CFC, que fora criado no ambiente do Microsoft Access. Neste ambiente será instanciado o *WScript*, onde os recursos da linguagem de *script* serão utilizados em conjunto com a linguagem VBA (*Visual Basic for Applications*). A figura 23 demonstra o processo realizado atualmente pelo Diretor de Ensino. O processo é executado de forma manual, sendo gerado um arquivo para cada instrutor.

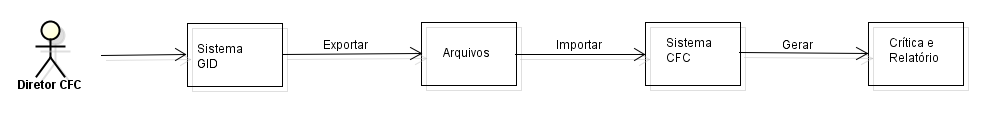


Figura 23 Processo de crítica atual

A figura 24 demonstra o novo processo, onde se utiliza a WSH e o programa *AutoHotkey*[[3]](#footnote-3). A WSH será utilizada para criar o *script* e executá-lo utilizando o programa *AutoHotkey*. O *AutoHotkey* será responsável pela execução dos comandos sobre a interface do GID.

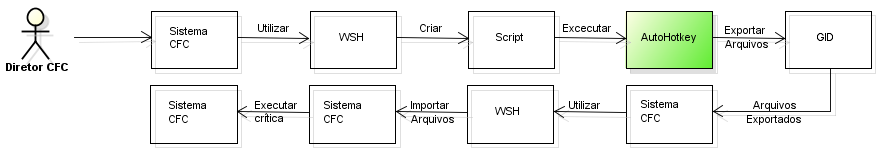


Figura 24 Processo de crítica novo utilizando WSH e AutoHotkey

## Utilização da WSH para integrar os sistemas CFC e GID

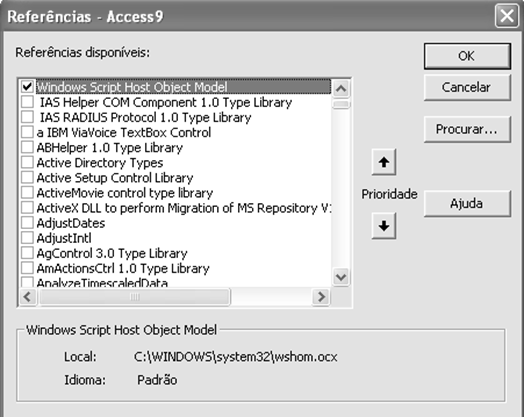
Este capítulo descreve como foi desenvolvido o script, as dificuldades encontradas e as soluções utilizadas. A primeira parate do capístulo destina-se a explicar os tipos de declarações.

### Tipos de declaração

A declaração da instância de um objeto em linguagens do padrão *VBA*, pode se dar de duas formas: *Late Binding* (Ligação Tardia) e *Early Binding* (Ligação Atual). As formas de declaração definem como o programa interage com o objeto no desenvolvimento e na execução. As diferenças entre as declarações são tratadas nos tópicos abaixo.

### Declaração do tipo *Early Binding*

O uso da declaração *Early Binding* se dá através da declaração explícita do objeto nas referências do ambiente de programação. A referência criada ao objeto WSH é demonstrada na figura 25, onde o item “A” da figura demonstra o nome da biblioteca, *Windows Script Host Object Model*, e o item “B” demonstra o local e o nome físico da biblioteca.



**B**

**A**

Figura 25 Declaração da Referência *Early Binding*

A vantagem em se utilizar a declaração *Early Binding* está no maior suporte ao desenvolvedor, possibilitando acesso a todas as instâncias do objeto com suporte a ajuda, assistente de sintaxe, além de maior controle na compilação e depuração. A desvantagem do seu uso é a necessidade que a mesma versão da biblioteca seja instalada em todos os micros onde a aplicação será utilizada. No caso de executar o sistema em máquinas que não tenham a mesma versão da WSH instalada, ocorrerá um erro na execução do sistema devido à falha na referência.

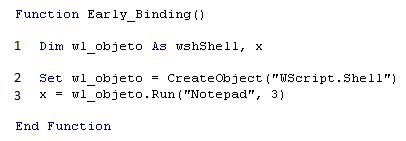


Figura 26 Declaração *Early Binding*

A figura 26 demonstra o uso do objeto *WSH* por *Early Binding*. Na linha 1 é criado um objeto do tipo wshShell, da biblioteca referenciada na figura 25, item “A”. A linha 2 demonstra a criação do objeto WSH, através do comando “SET”. O uso do objeto é demonstrado na linha 3. Com a utilização do método “Run” para executar o programa *notepad* (bloco de notas) do Windows.

### Declaração do tipo Late Binding

O uso da declaração por *Late Binding* é demonstrado na figura 27. Neste tipo de declaração não se realiza a referência à biblioteca, apenas se cria uma variável do tipo *object* (objeto) demonstrado na linha 1. Na linha 2, é definido que a variável do tipo objeto será um objeto do tipo WSH. Na linha 3 é utilizado o método “Run” para executar o programa bloco de notas.

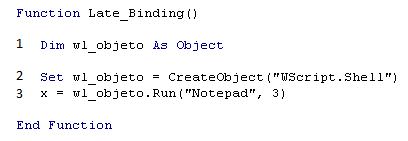


Figura 27 Declaração *Late Binding*

O uso da declaração *Late Binding* tem como vantagem o fato de não fazer uma referência explicita, ou seja, o objeto é instanciado de forma indireta, permitindo que sua execução seja efetuada independente da versão do objeto. A desvantagem reside no fato de requerer maior conhecimento de suas instâncias pelo desenvolvedor, não oferecer suporte no uso de suas instâncias, além de erros não serem detectados na compilação, ocorrendo somente em tempo de execução.

### Diferenças entre o uso da Late Binding e Early Binding

Ao analisar as duas formas de declaração, pode-se observar que a principal diferença está na declaração das referências e na declaração do tipo de variável. Na *Early Binding* declara-se explicitamente o objeto, considerando a sua versão e local de instalação, além de declarar a variável do tipo referenciado. A *Late Binding* faz referência a um objeto instanciado no sistema operacional. O programa não se preocupa com a versão e local de instalação da biblioteca, pois isto é de responsabilidade do sistema operacional.

Pode-se ainda utilizar o Microsoft Excel para exemplificar a diferença entre as duas formas de declaração. O Microsoft Excel é um programa que possui várias versões, onde cada versão possui sua própria biblioteca de objetos. Ao se criar uma referência utilizando o padrão *Early Binding*, o sistema será executado com sucesso somente se a biblioteca do Excel instalada nos computadores for a mesma referenciada nas propriedades do programa. Caso contrário, ocorrerá um erro em tempo de execução.

Ao utilizar a declaração do tipo *Late Binding* é efetuada a referência a um objeto do tipo Excel. Este tipo de declaração utiliza o objeto instanciado no sistema operacional, independente da versão ou local de uso. A vantagem desta forma de uso é tornar o programa compatível com as diferentes versões dos objetos instalados nos computadores. É importante ressaltar que a execução ocorrerá com sucesso somente se forem utilizados os métodos, propriedades e recursos comuns às versões do Excel instaladas.

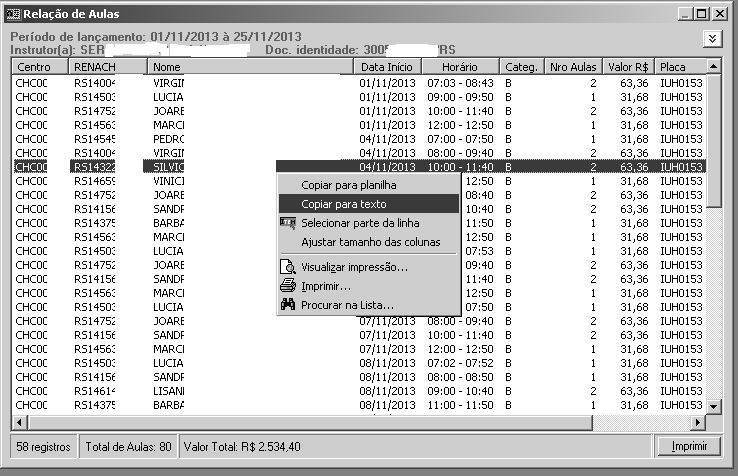
O cenário onde o sistema do CFC é utilizado é composto de micros com versões diferentes do Windows e, consequentemente, do *WSH*. Desta forma se optou em utilizar a declaração *Late Binding* possibilitando que o sistema seja executado em todos os micros independente da versão do *WSH*.

### Dificuldades encontradas

Durante o desenvolvimento do script, surgiram dificuldades. As principais dificuldades foram no desenvolvimento de rotina para controlar do mouse e o uso da WSH para interagir com a interface do GID

#### Controle do Mouse

Ao desenvolver o *script*, um dos problemas encontrados foi a necessidade de simular a ação do mouse pelo usuário, pois está é a única forma de exportar os dados dos instrutores no sistema GID. A figura 28 demonstra que não existe nenhum comando ou botão para efetuar a exportação dos dados da consulta. A única forma de exportar os mesmos, se dá através do uso do botão direito do mouse sobre o resultado da consulta (item A). Ao utilizar o mouse, é apresentado o menu de atalho (item B). A exportação se dá através do uso da opção “Copiar para texto” (demonstrado no item C).



**B**

**C**

**A**

Figura 28 Utilização do mouse para exportar os dados referente as aulas ministradas

Foi possível encontrar exemplos que utilizam as *API’s* do sistema operacional *Windows.* Elas permitem controlar o mouse, simulando a ação deste pelo usuário (MSDN, 2010). As *API’s* são bibliotecas do sistema operacional que permitem o acesso às funcionalidades do mesmo.

##### Gerenciamento do Mouse

O gerenciamento e controle do mouse ocorre através do uso das API*’s*, bibliotecas de funções do sistema operacional Windows. O uso das *API’s* é demonstrado a seguir.

##### *API SetCursorPos*

A *API* SetCursorPos é utilizada para posicionar o mouse em uma determinada coordenada na tela. Tem como parâmetros as coordenadas x e y. A coordenada x representa a coluna e y é utilizada para determinar a linha. A declaração da *API* é demonstrada abaixo.

Quadro 4 Declaração API para controlar a posição do cursor

|  |
| --- |
| Public Declare Function SetCursorPos Lib "user32" (ByVal x As Long, ByVal y As Long) As Long |

##### *API* mouse\_event

A *API* mouse\_event é utilizada para executar as ações realizadas pelo *mouse*. As ações que podem ser executadas são: o clique com o botão direito, clique com o botão esquerdo e o duplo clique. A declaração da API é demonstrada abaixo.

Quadro 5 Declaração API para controlar os eventod do *mouse*

|  |
| --- |
| Public Declare Sub mouse\_event Lib "user32" (ByVal dwFlags As Long, ByVal dx As Long, ByVal dy As Long, ByVal cButtons As Long, ByVal dwExtraInfo As Long) |

Para controlar os eventos do mouse foram criadas quatro variáveis públicas do tipo constante, demonstradas no quadro abaixo. A variável MOUSEEVENTF\_LEFTDOWN serve para determinar quando o botão esquerdo deve ser pressionado, a variável MOUSEEVENTF\_LEFTUP, é utilizada para executar a ação de soltar o botão esquerdo do mouse. As variáveis MOUSEEVENTF\_RIGHTDOWN e MOUSEEVENTF\_RIGHTUP servem para determinar quando o botão direito deve ser pressionado e quando deve ser liberado, respectivamente. A declaração das variáveis é apresentada no quadro abaixo.

Quadro 6 Declaração de variáveis para controle do *mouse*

|  |
| --- |
| Public Const MOUSEEVENTF\_LEFTDOWN = &H2  **Public Const MOUSEEVENTF\_LEFTUP = &H4**  Public Const MOUSEEVENTF\_RIGHTDOWN = &H8  Public Const MOUSEEVENTF\_RIGHTUP = &H10 |

O uso da *API* mouse\_event, foi dividido em funções públicas para cada tipo de ação a ser executada com o mouse. As funções Mouse\_Esquerdo\_click, Mouse\_Direito\_click e Mouse\_Duplo\_Click recebem os parâmetros wl\_col e wl\_lin para posicionar o mouse na coluna e linha da tela respectivamente. O funcionamento das funções é similar, recebem os parâmetros de coluna e linha, utilizam a API SetCursorPos para posicionar o ponteiro do mouse e utilizam a *API* mouse\_event para executar as ações do mouse. As três funções são apresentadas abaixo:

Quadro 7 Função para uso botão esquerdo do *mouse*

|  |
| --- |
| Public Function Mouse\_Esquerdo\_Click(wl\_col, wl\_lin)  SetCursorPos wl\_col, wl\_lin  mouse\_event MOUSEEVENTF\_LEFTDOWN, 0, 0, 0, 0  mouse\_event MOUSEEVENTF\_LEFTUP, 0, 0, 0, 0  End Sub |

Quadro 8 Função para uso botão direito do mouse

|  |
| --- |
| Private Sub Mouse\_Direito\_Click(wl\_col, wl\_lin)  SetCursorPos wl\_col, wl\_lin  mouse\_event MOUSEEVENTF\_RIGHTDOWN, 0, 0, 0, 0  mouse\_event MOUSEEVENTF\_RIGHTUP, 0, 0, 0, 0  End Sub |

Quadro 9 Função para uso duplo clique do mouse

|  |
| --- |
| Public Function Mouse\_Duplo\_Click (wl\_col, wl\_lin)  SetCursorPos wl\_col, wl\_lin  mouse\_event MOUSEEVENTF\_LEFTDOWN, 0, 0, 0, 0  mouse\_event MOUSEEVENTF\_LEFTUP, 0, 0, 0, 0  mouse\_event MOUSEEVENTF\_LEFTDOWN, 0, 0, 0, 0  mouse\_event MOUSEEVENTF\_LEFTUP, 0, 0, 0, 0  End Sub |

##### Exemplos de funções utilizando as APIs mouse\_event e SetCursorPos

A função demonstrada no quadro 10, apresenta o uso das funções mouse\_event e SetCursorPos para manipular o menu do programa Bloco de Notas. Nas linhas 1 e 2 são efetuadas as declarações da API mouse\_event e SetCursorPos respectivamente. Na linha 3 é declarada uma variável do tipo objeto. Na linha 5 o comando set é utilizado para determinar que a variável criada na linha 3 seja do tipo WSH.

Quadro 10 Exemplo de uso das funções SetCursorPos e mouse\_event

|  |
| --- |
| 1 Public Declare Sub mouse\_event Lib "user32" (ByVal dwFlags As Long, ByVal dx As Long,  ByVal dy As Long, ByVal cButtons As Long, ByVal dwExtraInfo As Long)  2 Public Declare Function SetCursorPos Lib "user32" (ByVal x As Long, ByVal y As Long) As Long  Function wsh\_exemplo\_notepad()  3 Dim wshShell As Object  4 Dim r As Long  5 Set wshShell = CreateObject("wscript.Shell")  6 x = wshShell.Run("notepad", 3)  7 Sleep (2000)  8 SetCursorPos 30, 38  9 mouse\_event MOUSEEVENTF\_LEFTDOWN, 0, 0, 0, 0  10 mouse\_event MOUSEEVENTF\_LEFTUP, 0, 0, 0, 0  11 Sleep (2000)  12 SetCursorPos 30, 88  13 mouse\_event MOUSEEVENTF\_LEFTDOWN, 0, 0, 0, 0  14 mouse\_event MOUSEEVENTF\_LEFTUP, 0, 0, 0, 0  15 wshShell.SendKeys ("teste de funçao")  16 Sleep (2000)  17 SetCursorPos 510, 420  18 mouse\_event MOUSEEVENTF\_LEFTDOWN, 0, 0, 0, 0  19 mouse\_event MOUSEEVENTF\_LEFTUP, 0, 0, 0, 0  End Function |

Na linha 6 é executado o comando *run* do objeto WSH. Este comando efetua a execução do programa Bloco de Notas. O comando Sleep(2000) utilizado na linha 7, 11 e 16, efetua uma pausa de 2 segundos durante a execução do programa. A pausa é necessária, pois cada comando tem um tempo de execução. Os tempos são diferentes para as ações executadas sobre comandos de preenchimento de campos dos comandos que executam uma consulta ou que alternem entre as janelas abertas no programa.

Na linha 8 é utilizado o comando SetCursorPos para posicionar o mouse na posição 30,38. Esta posição corresponde ao menu arquivo do programa Bloco de Notas. Na linha seguinte, é executado o clique do mouse através do comando mouse\_event, a fim de clicar sobre o menu. Sempre que utilizado o clique do mouse, é necessário usar um comando para pressionar o botão (MOUSEEVENTF\_LEFTDOWN) e outro para liberar o botão MOUSEEVENTF\_LEFTUP). Após executar o clique sobre o menu Arquivo, é executado o clique sobre a opção Salvar, processo demonstrado nas linhas 12, 13, 14.

Na linha 15 é utilizado o WSH para enviar o nome do arquivo para a caixa de diálogo salvar. Em seguida utiliza-se as funções da *API* para clicar sobre o botão salvar, demonstrado nas linhas 17, 18 e 19.

## Desenvolvimento da solução proposta

O *script* foi desenvolvido em um ambiente de testes, utilizando o Excel, para enviar os comandos do menu e controle do mouse. O Excel foi escolhido em virtude de o programa GID só poder ser utilizado no ambiente do CFC. Além deste impeditivo, os testes só podem ser realizados no CFC em horário que o diretor de ensino se encontra, pois o arquivo com as informações das aulas ministradas só pode ser gerado através do seu usuário.

Além do uso do Excel, foram realizados vários testes, inclusive com o uso do *script* aplicado em outros programas (Microsoft Word, Winrar e o Bloco de Notas), todos com sucesso. O *script* de conexão com o GID é demonstrado no quadro abaixo. As explicações serão descritas em partes. O código completo encontra-se no apêndice sob o título “Código fonte para gerar *Script* integração GID, utilizando a WSH”.

O código abaixo posiciona o foco no sistema GID. O comando *sleep* é utilizado para que o sistema aguarde um tempo, armazenado na variável wl\_tempo\_programa, entre o posicionamento e a execução da próxima instrução. A chamada de função call MouseSingle, é utilizada para executar a ação do mouse sobre a opção de menu do sistema GID.

Quadro 11 WSH - Posicionando o Foco no sistema GID

|  |
| --- |
| wl\_wsh\_shell.AppActivate ("GID")  Sleep (wl\_tempo\_programa)  Call MouseSingleClick(151, 58)  Sleep (wl\_tempo\_comando) |

As ações abaixo descritas, são utilizadas para preencher os campos para a pesquisa no sistema GID. O sistema GID abre a janela de pesquisa com os dados da última consulta realizada. As linhas 1 a 4 são utilizadas para excluir estes dados. O preenchimento dos campos é demonstrado das linhas 5 a 14. A linha 15 é utilizada para executar a consulta através do envio do ALT+L, atalho do botão executar no GID.

Quadro 12 WSH - Preenchimento dos campos da tea de consulta do sistema GID

|  |
| --- |
| 1 wl\_wsh\_shell.SendKeys ("{DEL}")  2 Sleep (wl\_tempo\_comando)  3 wl\_wsh\_shell.SendKeys ("{DEL}")  4 Sleep (wl\_tempo\_comando)  5 wl\_wsh\_shell.SendKeys (!inscr\_icmsrg)  6 Sleep (wl\_tempo\_comando)  7 wl\_wsh\_shell.SendKeys ("{TAB}")  8 Sleep (wl\_tempo\_comando)  9 wl\_wsh\_shell.SendKeys (wl\_dt\_ini)  10 Sleep (wl\_tempo\_comando)  11 wl\_wsh\_shell.SendKeys ("{TAB}")  12 Sleep (wl\_tempo\_comando)  13 wl\_wsh\_shell.SendKeys (wl\_dt\_fim)  14 Sleep (wl\_tempo\_comando)  15 wl\_wsh\_shell.SendKeys ("%L")  16 Sleep (wl\_tempo\_comando) |

As ações do quadro a seguir, servem para abrir o menu de contexto do mouse, linha 1, executar a opção “copiar texto”, linhas 3, 5 e 7. A linha 9 é utilizada para fechar a janela de pesquisa do GID.

Quadro 13 Salvar os dados em arquivo texto

|  |
| --- |
| 1 Call MouseSingleClick(373, 211)  2 Sleep (wl\_tempo\_programa)  3 wl\_wsh\_shell.SendKeys ("{DOWN}")  4 Sleep (wl\_tempo\_comando)  5 wl\_wsh\_shell.SendKeys ("{DOWN}")  6 Sleep (wl\_tempo\_comando)  7 wl\_wsh\_shell.SendKeys ("{ENTER}")  8 Sleep (wl\_tempo\_comando)  9 wl\_wsh\_shell.SendKeys ("%{F4}")  10 Sleep (wl\_tempo\_comando) |

A parte do código descrita a seguir é responsável pela exportação dos dados copiados para o programa Bloco de Notas. A abertura do programa é demonstrada na linha 1. A linha 2 executa a opção CTRL+V. A linha 5 demonstra o acesso ao menu do Bloco de Notas e a utilização da opção da opção “Salvar Como”. A linha 9 demonstra o preenchimento do campo nome do arquivo, com o nome da pasta e o nome do arquivo. A ação salvar é executada através do uso do atalho do botão Salvar (linha 11). O programa bloco de notas é fechado através das linhas 13, 14 e 15.

Quadro 14 Executar ação salvar no programa Bloco de Notas

|  |
| --- |
| 1 x = wl\_wsh\_shell.Run("notepad", 3)  2 Sleep (wl\_tempo\_programa)  3 wl\_wsh\_shell.SendKeys ("^v")  4 Sleep (wl\_tempo\_comando)  5 wl\_wsh\_shell.SendKeys ("%A")  6 Sleep (wl\_tempo\_comando)  7 wl\_wsh\_shell.SendKeys ("%V")  8 Sleep (wl\_tempo\_comando)  9 wl\_wsh\_shell.SendKeys (wl\_pasta\_script & wl\_arq)  10 Sleep (wl\_tempo\_comando)  11 wl\_wsh\_shell.SendKeys ("%L")  12 Sleep (wl\_tempo\_comando)  13 wl\_wsh\_shell.SendKeys ("%A")  14 Sleep (wl\_tempo\_comando)  15 wl\_wsh\_shell.SendKeys ("%S") |

O quadro abaixo demonstra a verificação se o arquivo com os dados do instrutor foi criado com sucesso (linha 16). Caso o mesmo não tenha sido criado, é apresentada mensagem ao usuário questionando se deseja executar o *script* para o instrutor novamente (linhas 16 a 22).

Quadro 15 *Scrit* para testar se o arquivo foi criado

|  |
| --- |
| 16 If Dir(wl\_pasta\_script & wl\_arq) = "" Then  17 If MsgBox("Arquivo não foi gerado para o instrutor: " & vbCrLf & vbCrLf & !razao\_func & vbCrLf & vbCrLf & "Deseja tentar novamente?", vbQuestion + vbYesNo, "Atenção !!!") = vbYes Then  18 Resume gerar\_arq\_gid  19 Else  20 Resume Exit\_Bas\_Gid\_Integracao  21 End If  22 End If |

## Testes no ambiente do CFC

Os testes de integração entre os sistemas CFC e GID, através da utilização do *script*, apresentaram erros durante a sua execução. Os comandos para alternar entre os sistemas funcionaram perfeitamente. Já os comandos enviados para simular o uso do teclado e mouse não obtiveram sucesso.

A falha na execução do *script* sobre a interface do GID está relacionada ou a plataforma em que o mesmo foi desenvolvido ou a um tipo de bloqueio quanto ao recebimento de teclas através do comando *sendkeys*. A solução foi efetuar uma pesquisa buscando uma solução para o envio de comandos. Foi encontrada a ferramenta *AutoHotkey*, explicada com maiores detalhes no tópico “Ferramenta AutoHotKey”. É possível utilizar esta ferramenta em conjunto com a WSH, para solucionar o problema do envio de comandos. A WSH é utilizada para criar o arquivo contendo o *script* e executá-lo utilizando o programa *AutoHotkey*. O programa *AutoHotkey* será o responsável por enviar as teclas de controle de menu e mouse para o sistema GID.

##### Ferramenta AutoHotkey

O programa AutoHotkey foi desenvolvido para possibilitar a criação de *scripts* voltados para a automatização de tarefas. As principais características de acordo com AutoHotkey (2014) são:

* Criar atalhos para teclado, *joystick* e *mouse*.
* Criar formulários personalizados de entrada de dados, interfaces de usuário, e barras de menu.
* Remapear teclas e botões do teclado, *joystick* e *mouse*.
* Responder a sinais de controles remotos de mão através do *script* de cliente WinLIRC[[4]](#footnote-4).
* Executar *scripts* criados com AutoIt[[5]](#footnote-5) v2
* Converter *script* em um arquivo EXE que pode ser executado em computadores que não possuem AutoHotkey instalado.

Os *scripts* podem ser criados utilizando um editor de textos sem formatação. Após a criação do *script*, o mesmo pode ser executado de duas formas. A primeira forma é através da execução pela linha de comando. A segunda forma é abrir o programa na memória e abrir o *script* para executar.

A execução do *script*, através da linha de comando, possibilita que a ferramenta *AutoHotkey* seja executada por outros programas. A sintaxe para execução é: *AutoHotkey* <arquivo\_script>, onde arquivo\_*script* deve ser substituído pelo nome do arquivo a ser executado.

O exemplo demonstrado no quadro abaixo apresenta um *script* utilizado para abrir o programa notepad (bloco de notas). O funcionamento do *script* consiste em verificar se o programa está aberto na memória, caso esteja aberto, o foco é movido para o mesmo. Caso contrário, o programa é aberto e após, o foco é movido para o mesmo.

Quadro 16 Exemplo de script para abrir o programa Bloco de Notas

|  |
| --- |
| IfWinExist Sem título - Notepad  {  WinActivate  }  else  {  Run Notepad  WinWaitActive Sem título - Notepad  WinActivate  } |

Os comandos utilizados no *script* são explicados de forma mais detalhada abaixo:

- IfWinActive: verificar se a janela especifica está atualmente ativa.

- WinWaitActive: alterna o foco para a janela especificada.

- WinWait: Aguarda a janela receber o foco.

- Run: executa o programa.

##### Arquivo de *script* criado pela WSH para execução na ferramenta *AutoHotkey*

Neste tópico serão explicadas as partes mais importantes do *script*. O código completo do mesmo está disponibilizado no apêndice sob o título “Código fonte para gerar Script integração GID”.

O quadro abaixo demonstra o uso do comando WinWait para posicionar o foco no sistema GID, linha 1. Na linha 2 é utilizado o comando WinWaitActive que é responsável por aguardar até que a janela do GID receba o foco. Nas linhas 3 e 5 é demonstrado o uso do botão esquerdo do mouse nas coordenadas horizontal e vertical respectivamente, para abrir a janela de pesquisa de aulas lançadas no sistema GID. A linha 6 é utilizada para fazer uma pausa no *script* para que o comando seja executado.

Quadro 17 Uso do comando WinWait

|  |
| --- |
| 1 wl\_string\_cab = "WinWait, GIDHab [CHC00302]," & vbCrLf & \_  "IfWinNotActive, GIDHab [CHC00302], , WinActivate, GIDHab [CHC00302]," & vbCrLf & \_  2 "WinWaitActive, GIDHab [CHC00302]," & vbCrLf & \_  3 "MouseClick , Left, 505, 32" & vbCrLf & \_  4 "Sleep , " & wl\_tempo\_comando & vbCrLf & \_  5 "MouseClick , Left, 543, 94" & vbCrLf & \_  6 "Sleep , " & wl\_tempo\_comando |

O código abaixo é utilizado para preencher os campos da janela de pesquisa do sistema GID com as informações do período a ser consultado e com o RG do instrutor (linha 7). A linha 10 tem os comandos para mover o foco sobre o botão “Localizar”. A linha 9 demonstra o uso do botão direito do mouse do mouse e a linha 11 aciona a 2ª opção do menu. A linha 13 é utilizada para abrir a caixa de diálogo executar do sistema operacional Windows enquanto que a linha 18 demonstra a abertura do programa Bloco de Notas.

Quadro 18 Preenchimento dos campos da janela de pesquisa do GID

|  |
| --- |
| 1 wl\_arq\_script.writeline "WinWait, Relação de Aulas,"  2 wl\_arq\_script.writeline "IfWinNotActive, Relação de Aulas, , WinActivate, Relação de Aulas,"  3 wl\_arq\_script.writeline "WinWaitActive, Relação de Aulas,"  4 wl\_arq\_script.writeline "MouseClick , Left, 151, 58"  5 wl\_arq\_script.writeline "MouseClick , Left, 151, 58"  6 wl\_arq\_script.writeline "Sleep , " & wl\_tempo\_comando  7 wl\_arq\_script.writeline "Send, {DEL}" & wl\_rg\_func & "{TAB}rs{TAB}" & Format(Me.dt\_ini, "ddmmyyyy") & "{TAB}" & Format(Me.dt\_fim, "ddmmyyyy") & "{ALTDOWN}l{ALTUP}"  8 wl\_arq\_script.writeline "Sleep , " & wl\_tempo\_programa  9 wl\_arq\_script.writeline "MouseClick , Right, 373, 211”  10 wl\_arq\_script.writeline "Sleep , " & wl\_tempo\_programa  11 wl\_arq\_script.writeline "Send, {DOWN}{DOWN}{ENTER}"  12 wl\_arq\_script.writeline "Sleep , " & wl\_tempo\_programa  13 wl\_arq\_script.writeline "Send, {LWINDOWN}r{LWINUP}"  14 wl\_arq\_script.writeline "Sleep , " & wl\_tempo\_comando  15 wl\_arq\_script.writeline "WinWait, Executar,"  16 wl\_arq\_script.writeline "IfWinNotActive, Executar, , WinActivate, Executar,"  17 wl\_arq\_script.writeline "WinWaitActive, Executar,"  18 wl\_arq\_script.writeline "Send, notepad{ENTER}"  19 wl\_arq\_script.writeline "Sleep, " & wl\_tempo\_programa |

No código abaixo é verificado se o programa bloco de notas foi aberto com sucesso, linha 2 e em seguida é executado o comando colar, linha 4. Na linha 6 é enviado o comando “salvar como” para o Bloco de Notas. O comando para determinar o caminho onde será salvo o arquivo e o nome do mesmo é demonstrado nas linhas 8 e 10. O programa Bloco de Notas é encerrado com o comando demonstrado na linha 11.

Quadro 19 Execução do comando salavar no programa Bloco de Notas

|  |
| --- |
| 1 wl\_arq\_script.writeline "WinWait, Sem título - Bloco de notas,"  2 wl\_arq\_script.writeline "IfWinNotActive, Sem título - Bloco de notas, , WinActivate, Sem título - Bloco de notas,"  3 wl\_arq\_script.writeline "WinWaitActive, Sem título - Bloco de notas,"  4 wl\_arq\_script.writeline "Send, {ALTDOWN}{ALTUP}el{ALTDOWN}{ALTUP}av"  5 wl\_arq\_script.writeline "WinWait, Salvar como,"  6 wl\_arq\_script.writeline "IfWinNotActive, Salvar como, , WinActivate, Salvar como,"  7 wl\_arq\_script.writeline "WinWaitActive, Salvar como,"  8 wl\_arq\_script.writeline "Send, " & wl\_pasta\_script & wl\_rg\_func & "{ALTDOWN}l{ALTUP}"  9 wl\_arq\_script.writeline "WinWait, " & wl\_arq & " - Bloco de notas,"  10 wl\_arq\_script.writeline "IfWinNotActive, " & wl\_arq & " - Bloco de notas, , WinActivate, " & wl\_arq & " - Bloco de notas,"  11 wl\_arq\_script.writeline "Send, {ALTDOWN}{F4}{ALTUP}" |

No quadro abaixo, o comando da linha 1 encerra a criação do *script*. As linhas 2 e 3 executam o arquivo criado pela WSH através do programa *AutoHotkey*.

Quadro 20 Execução do programa AutoHotkey através da WSH

|  |
| --- |
| 1 wl\_arq\_script.Close  2 wl\_string = wl\_pasta\_autohotkey & "AutoHotkey.exe " & wl\_pasta\_script & "gid.txt"  3 wl\_wsh\_shell.Run wl\_string |

## Resultados dos testes realizados *in loco*

A execução dos testes foi dividida em 2 etapas, ambas realizadas no ambiente do CFC sob a supervisão do Diretor de Ensino. A primeira etapa dos testes, foi efetuada para validar o código de programação, criação do *script*, a execução do mesmo com a linguagem WSH e a efetividade do programa *AutoHotkey*. Esta etapa serviu para identificar os problemas na execução do *script* a fim de gerar uma versão de testes para ser executada pelo Diretor de Ensino.

Durante a execução dos testes ocorreu um erro ao salvar os arquivos textos com os dados. Este erro foi causado porque o comando de copiar os dados era executado antes dos dados serem disponibilizados. Este erro ocorreu devido ao tempo da consulta ser maior que o tempo previsto para execução do próximo comando. Desta forma, foram realizados vários testes para identificar os tempos necessários entre a execução dos comandos. Como o script roda sobre a interface, não tem como saber o tempo que a consulta irá levar.

Foi necessário criar três tempos diferentes para a execução dos comandos. Um tempo para o intervalo entre os comandos da interface, um segundo para o intervalo entre a consulta aos dados e o comando seguinte e o terceiro tempo para gerenciar a alternância entre o sistema GID e o programa Boco de Notas. Para possibilitar uma maior gerência dos tempos, foi criada uma tela para parametrizar os tempos, demonstrado na figura 29, desta forma, é possível fazer ajustes conforme o desempenho do sistema GID e do computador pelo próprio Diretor de Ensino.

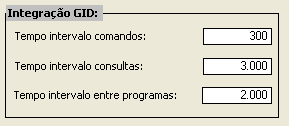
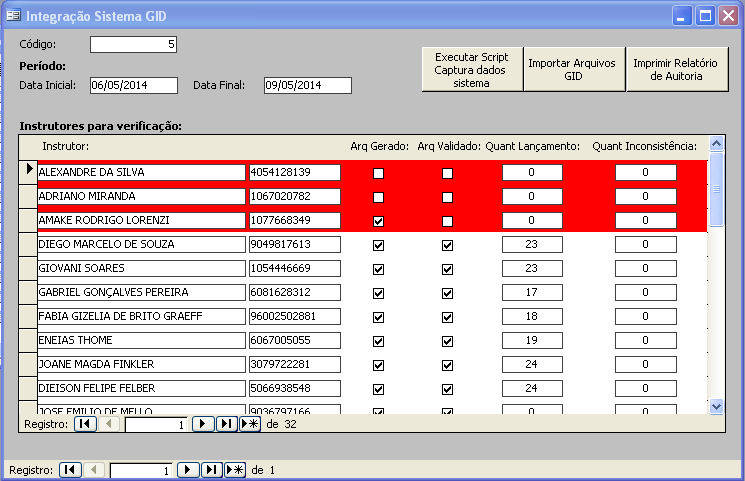


Figura 29 Tela de Parâmetros dos Tempos

Durante os testes, foi necessário criar uma rotina para identificar quais arquivos não foram gerados e quais não atenderam aos critérios da pesquisa. A solução proposta foi a de organizar os instrutores com problemas na integração sempre no topo da lista com um fundo vermelho (Item A da figura 29). O item B da figura demonstra quais foram os instrutores que não tiveram o arquivo exportado pelo GID. O item C é utilizado para indicar se estava de acordo com os critérios da pesquisa.



**C**

**B**

**A**

Figura 30 Resultado da integração

A segunda etapa dos testes foi destinada para execução pelo Diretor de Ensino. Foi apresentado ao mesmo um roteiro[[6]](#footnote-6) para executar a integração e efetuar a demonstração do processo. O Diretor de Ensino realizou quatro testes de integração seguindo o roteiro. Para validar os resultados, os mesmos foram comparados com os relatórios gerados pelo processo antigo. Os resultados demonstraram que a integração ocorreu com sucesso.

Segundo o Diretor de Ensino a integração apresentou três vantagens em relação ao processo antigo. A primeira vantagem foi a redução do tempo utilizado para a execução da crítica. No processo antigo, a critica demorava em torno de uma hora. Com o processo novo, a mesma foi realizada em 15 minutos. A segunda vantagem foi a de possibilitar ao Diretor de Ensino realizar outras atividades enquanto a rotina automatizada é executada. A última vantagem é a de realizar a auditoria de dois em dois dias ao invés de uma vez a cada 7 dias. Esta última vantagem representa um diferencial muito grande para o Diretor, pois permite monitorar de forma mais seguida as aulas ministradas e o lançamento das mesmas no sistema GID, propósito principal da auditoria.

CONCLUSÃO

Neste trabalho foi conceituado o que é a integração de sistemas, sua importância para o gerenciamento e qualidade da informação para as empresas. No cenário atual é comum as empresas utilizarem vários sistemas para atender a sua demanda organizacional. A integração visa reduzir o tempo gasto na inserção de dados, aumentar a confiabilidade da informação e propiciar uma vantagem competitiva.

A análise do problema de uso de dois sistemas pelo CFC, demonstrou uma situação onde o CFC não tem um controle diário sobre o que planeja e o que realiza. O planejamento e a cobrança das aulas são gerenciadas pelo sistema CFC, já o lançamento das aulas efetivas são realizadas no sistema GID. O problema para o CFC está na demora em auditar os dados informados no sistema CFC e no DETRAN. Esta demora faz com que a auditoria seja realizada somente uma vez por semana. Este trabalho reduziu o tempo gasto na auditoria, possibilitando a sua realização diária. O principal benefício para o CFC é o de ter um maior controle sobre o serviço dos instrutores e identificar possíveis irregularidades.

Visando solucionar o problema proposto, foi criado um *script* para acessar o sistema GID, a fim de gerar de forma automática os dados para realização da auditoria entre os sistemas. O *script* foi criado utilizando a linguagem WSH, descrita no capítulo 3, e o programa AutoHotkey descrito no capitulo 4.3. Além do desenvolvimento do *script* foi executada a implantação dele em um CFC, realizou-se os testes de execução e foi efetuada a documentação e correção de possíveis erros.

O principal objetivo do trabalho foi possibilitar a realização da auditoria diariamente, onde além de se reduzir o tempo gasto, procura-se possibilitar um maior gerenciamento das informações geradas nos dois sistemas. Outra questão muito importante é a apuração das inconsistências, possibilitando ao CFC agir de acordo com as normas determinadas pelo DETRAN.

O desenvolvimento do trabalho demonstrou que a linguagem WSH possui muitos recursos. Esta linguagem permite automatizar várias tarefas no sistema operacional Windows, possibilita executar programas, efetuar consultas em banco de dados, manipular programas, dentre outras tarefas. Embora o objetivo inicial deste trabalho, era de utilizar somente a linguagem WSH, não foi possível utilizar somente ela para interagir com a interface gráfica do GID. Foi necessário efetuar pesquisa a fim de encontrar uma solução ao problema. A solução encontrada foi o uso do programa *AutoHotkey* em conjunto com a linguagem WSH. A linguagem WSH foi utilizada para criar o arquivo de *script*, executar o programa *AutoHotkey* e para manipular os arquivos gerados. O programa *AutoHotkey* foi utilizado para efetuar a interação sobre a interface do GID.

A pesquisa sobre o programa *AutoHotkey* possibilitou ampliar o conhecimento sobre a automatização de processos no ambiente operacional Windows. A versatilidade do programa possibilita a criação de *script* interpretado e executável. Os *scripts* podem ser criados para realizar tarefas do sistema operacional e automatizar rotinas executadas em sistemas. O uso do *script* executável permite enviar o mesmo para execução em outros computadores.

É importante fazer a ressalva que a forma de integração utilizada neste trabalho, não é a melhor solução. Existem soluções como XML e WebServices, por exemplo, que possibilitam o total controle do fluxo de dados envolvidos. Porém, como não era possível utilizar nenhuma destas opções, a solução utilizada mostrou ser possível criar soluções para atender as necessidades específicas dos usuários do CFC. Mesmo quando se apresentaram situações incontroláveis, como o travamento do sistema GID ou a mudança de versão, foi possível utilizar a solução desenvolvida.

O desenvolvimento do trabalho atingiu os objetivos esperados pelo Diretor de Ensino do CFC. A implantação tornará possível realizar a auditoria diariamente, possibilitando um maior controle sobre as atividades realizadas pelos instrutores. Com a execução diária, será possível tomar medidas imediatas quando a auditoria apresenta os erros realizados pelos instrutores. Outro fator importante da integração foi possibilitar ao diretor de Ensino realizar outras tarefas enquanto o processo é realizado.

Como projeto futuro, fica o desafio de substituir o programa autohotkey por uma solução desenvolvida. A sugestão de se criar um componente utilizando a linguagem de programação Java, através do uso da classe Robot, permite maior controle na distribuição da solução além de evitar a utilização de um programa que pode ser descontinuado e/ou não ter suporte a versões futuras do Windows. Outra melhoria importante é criação de um mecanismo para consultar somente os instrutores cujo arquivo não foi criado e/ou não validado.

Referências Bibliográficas

A INTEGRAÇÃO como vantagem competitiva. Quidgest Lisboa, RS: Quidgest, Consultores de Gestão Ltda., dez. 2006.

AUTOHOTKEY. **AutoHotKey Automation, Hotkeys, Scripting**. Disponível em: < http://www.autohotkey.com/>. Acesso em: 6 out. 2014.

BATTISTI, Julio. **Introdução ao Windows Script Host**, 2005. Disponível em: <http://www.juliobattisti.com.br/tutoriais/ricardosilva/wsh001.asp>. Acesso em: 9 out. 2013.

BORN, Günter. **Microsoft Windows Script Host 2.0 Developer's Guide**. Washington: Microsoft Press, 2000.

COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim. **Sistemas Distribuídos Conceitos e Projeto**. 4 ed. São Paulo: Artmed Editora S. A., 2007.

DETRAN - DEPARTAMENTO ESTADUAL DE TRÂNSITO DO RS. **Sobre o DETRAN/RS**, 2013a. Disponível em: <http://www.DETRAN.rs.gov.br/index.php?action=portal&subm=5& cod=1>. Acesso em: 06 set. 2013.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. **Portaria/DETRAN-RS n° 70, de 13 de maio de 2002**, 2013b. Disponível em: <http://www.DETRAN.rs.gov.br/index. php? action=pub&codleg=1304&semBanner=1>. Acesso em: 20 ago. 2013.

FAERMAN, Julio. **Integração com JBI**, 2013. Disponível em: < http://www.devmedia.com.br/artigo-java-magazine-32-integracao-com-jbi/8984 >. Acesso em: 1 dez. 2013.

FORD, Jerry Lee, Jr. ***Windows Script Host***. 3 ed. Boston: Course Technology, 2008.

LAUDON, Kenneth C.; LAUDON, Jane P. **Sistemas de Informações Gerenciais**. 9 ed. São Paulo: Pearson Prentice Hall, 2010. Tradução de: Luciana do Amaral.

MARTINS, Victor. Integração de Sistemas de Informação: Perspectivas, normas e abordagens. Universidade do Minho, 1 ed. Portugal: Sílabo 2006

MENDES, Francisco Coelho. **Administração de Sistemas de Informação**. Rio de Janeiro: Fundação CECIERJ, 2009

MSDN Microsoft. **Windows Script Host Version Information**, 2009a. Disponível em: <http://msdn.microsoft.com/en-us/library/x66z77t4%28v=vs.84%29.aspx>. Acesso em: 2 out. 2013.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. **Reference (Windows Script Host)**, 2009b. Disponível em: <http://msdn.microsoft.com/en-us/library/98591fh7%28v=vs.84%29.aspx>. Acesso em: 2 out. 2013.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. **Windows Script Anfitrião Object Model**, 2009c. Disponível em: <http://msdn.microsoft.com/en-us/library/ms710252(v=vs.85).aspx>. Acesso em: 1 dez. 2013.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. **Mouse Operations**, 2010. Disponível em: <http://social.msdn.microsoft.com/Forums/vstudio/en-US/14ef4de0-037f-4f9f-bddc-27b5e94299f6/mouse-operations?forum=vbgeneral >. Acesso em: 1 mar. 2014.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. **Microsoft Open Database Connectivity (ODBC)**, 2013. Disponível em: <http://msdn.microsoft.com/en-us/library/a74hyyw0.aspx>. Acesso em: 2 out. 2013.

NETO, Agostinho Campos. **Web Services em Java com Axis - Teoria e Prática**, 2013. Disponível em: < http://www.aprendajavafacil.com.br/portal/ws.pdf >. Acesso em: 1 dez. 2013.

Oracle. **Lesson: JDBC Introduction**, 2013. Disponível em: <http://docs.oracle.com/ javase/tutorial/jdbc/overview/index.html>. Acesso em: 1 dez. 2013.

PAYNE, Eric. **Consultar um banco de dados Oracle**, 2009. Disponível em: <http://gallery.technet.microsoft.com/scriptcenter/24f03ee2-b0fe-48cc-83ca-de4e5520ba8f>. Acesso em: 9 out. 2013.

PRODANOV, Cleber; FREITAS, Ernani. **Metodologia do Trabalho Científico: Métodos e Técnicas da Pesquisa e do Trabalho Acadêmico**. 2ª ed. Novo Hamburgo: FEEVALE, 2013.

SINGH, Harry, S. **Data Warehouse: Conceitos, tecnologias, implementação e gerenciamento**. 1ª ed São Paulo: Makron Books, 2001.

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

SOUZA, Cesar Alexandre de (org.); SACCOL, Amarolinda Zanela (org.). **Sistemas ERP no Brasil**. São Paulo: Editora Atlas, 2003.

SPACKMAN, Devin; SPEAKER, Mark. **Soluções de Integração Empresarial**. Porto Alegre: Bookman, 2006. Tradução de: João Tortello.

TURBAN, Efrain; MCLEAN, Ephraim; WETHERBE, James. **Tecnologia da Informação para Gestão**. 3 ed. Porto Alegre: Bookman, 2004. Tradução de: Renata Schinke.

VEIGA, Roberto G. A. ***Windows Script Host***. São Paulo: Editora Novatec, 2000.

VISIGENIC SOFTWARE, Inc.. ***ODBC 2.0 Programmer’s Manual***. California: TimesTen Performance Software, 2000

apendíces

## Código fonte para gerar *Script* integração GID utilizando a WSH

Function bas\_Gid\_Integracao()

On Error GoTo Err\_Bas\_Gid\_Integracao

Dim wl\_dbf As Database, \_

wl\_dyn\_func As Recordset, \_

wl\_tbl\_gid\_integracao\_ocorrencia As Recordset, \_

wl\_wsh\_shell As Object, \_

wl\_pasta\_script, wl\_arq, wl\_arq\_script, wl\_sql, \_

wl\_tempo\_comando, wl\_tempo\_programa, r, wl\_dt\_ini, wl\_dt\_fim

Set wl\_wsh\_shell = CreateObject("wscript.Shell")

wl\_tempo\_comando = 500

wl\_tempo\_programa = 2000

Set wl\_dbf = OpenDatabase(wg\_sis\_arq\_dbf)

wl\_sql = "SELECT tbl\_func.cod\_func, tbl\_func.razao\_func, tbl\_func.dt\_cadastro, tbl\_func.inscr\_icmsrg " & \_

"FROM tbl\_func " & \_

"WHERE exists (select id\_instrutor from tbl\_pratica where tbl\_pratica.id\_instrutor=tbl\_func.cod\_func and dt\_pratica>=#" & Format(Me.dt\_ini, "mm/dd/yy") & "# and dt\_pratica<=#" & Format(Me.dt\_fim, "mm/dd/yy") & "#) " & \_

"ORDER BY razao\_func"

Set wl\_dyn\_func = CurrentDb.OpenRecordset(wl\_sql, dbOpenDynaset)

Set wl\_tbl\_gid\_integracao\_ocorrencia = wl\_dbf.OpenRecordset("tbl\_gid\_integracao\_ocorrencia", dbOpenTable)

wl\_tbl\_gid\_integracao\_ocorrencia.Index = ""

' wg\_sis\_arq\_imp = !pasta\_arqs

' wg\_sis\_arq\_bkp = !pasta\_backup

wl\_pasta\_script = wg\_sis\_arq\_imp

With wl\_dyn\_func

Do Until .EOF

gerar\_arq\_gid:

wl\_arq = !razao\_func & ".txt"

If Dir(wl\_pasta\_script & wl\_arq) <> "" Then

Kill wl\_pasta\_script & wl\_arq

End If

wl\_wsh\_shell.AppActivate ("GID")

Sleep (wl\_tempo\_programa)

Call MouseSingleClick(151, 58)

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("{DEL}")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("{DEL}")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys (!inscr\_icmsrg)

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("{TAB}")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys (wl\_dt\_ini)

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("{TAB}")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys (wl\_dt\_fim)

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("%L")

Sleep (wl\_tempo\_comando)

Call MouseSingleClick(373, 211)

Sleep (wl\_tempo\_programa)

wl\_wsh\_shell.SendKeys ("{DOWN}")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("{DOWN}")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("{ENTER}")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("%{F4}")

Sleep (wl\_tempo\_comando)

x = wl\_wsh\_shell.Run("notepad", 3)

Sleep (wl\_tempo\_programa)

wl\_wsh\_shell.SendKeys ("^v")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("%A")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("%V")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys (wl\_pasta\_script & wl\_arq)

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("%L")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("%A")

Sleep (wl\_tempo\_comando)

wl\_wsh\_shell.SendKeys ("%S")

If Dir(wl\_pasta\_script & wl\_arq) = "" Then

If MsgBox("Arquivo não foi gerado para o instrutor: " & vbCrLf & vbCrLf & !razao\_func & vbCrLf & vbCrLf & "Deseja tentar novamente?", vbQuestion + vbYesNo, "Atenção !!!") = vbYes Then

Resume gerar\_arq\_gid

Else

Resume Exit\_Bas\_Gid\_Integracao

End If

End If

wl\_dyn\_func.MoveNext

Loop

End With

MsgBox "Arquivos gerados.", vbInformation, "Atenção !!!"

Exit\_Bas\_Gid\_Integracao:

Exit Sub

Err\_Bas\_Gid\_Integracao

If Err > 0 Then

MsgBox "Ocorreu o erro:" & vbCrLf & vbCrLf & Err & " - " & Error, vbInformation, "Atenção !!!"

Resume Exit\_Bas\_Gid\_Integracao

End If

End Function

## Código fonte para gerar *Script* integração GID utilizando a WSH e o AutoHotkey

Private Sub btn\_executar\_script\_Click()

On Error GoTo Err\_btn\_executar\_script\_Click

Dim wl\_dbf As Database, \_

wl\_dyn\_func As Recordset, \_

wl\_dyn As Recordset, \_

wl\_tbl\_gid\_integracao\_ocorrencia As Recordset, \_

wl\_dyn\_parametro As Recordset, \_

wl\_wsh\_shell As Object, \_

wl\_wsh\_fso As Object, \_

wl\_pasta\_script, wl\_pasta\_autohotkey, wl\_pasta\_integracao\_gid, wl\_arq, wl\_arq\_script, wl\_sql, \_

wl\_string, wl\_string\_cab, wl\_tempo\_comando, wl\_tempo\_programa, \_

wl\_razao\_func, wl\_rg\_func, wl\_cod\_gid\_integracao, wl\_qtd\_tentativa, wl\_cont\_instr

If IsNull(Me.dt\_ini) Or IsNull(Me.dt\_fim) Then

MsgBox "É necessário informar a data inicial e final.", vbInformation, "Atenção !!!"

Exit Sub

End If

wl\_cod\_gid\_integracao = Me.cod\_gid\_integracao

Set wl\_dbf = OpenDatabase(wg\_sis\_arq\_dbf)

wl\_sql = "SELECT tbl\_func.cod\_func, tbl\_func.razao\_func, tbl\_func.dt\_cadastro, tbl\_func.inscr\_icmsrg " & \_

"FROM tbl\_func " & \_

"WHERE exists (select id\_instrutor from tbl\_pratica where tbl\_pratica.id\_instrutor=tbl\_func.cod\_func and dt\_pratica>=#" & Format(Me.dt\_ini, "mm/dd/yy") & "# and dt\_pratica<=#" & Format(Me.dt\_fim, "mm/dd/yy") & "#) " & \_

"ORDER BY razao\_func"

Set wl\_dyn\_func = CurrentDb.OpenRecordset(wl\_sql, dbOpenDynaset)

Set wl\_dyn\_parametro = CurrentDb.OpenRecordset("tbl\_parametros", dbOpenDynaset)

wl\_tempo\_comando = wl\_dyn\_parametro!tempo\_comando\_integracao\_gid

wl\_tempo\_programa = wl\_dyn\_parametro!tempo\_programa\_integracao\_gid

wl\_pasta\_autohotkey = wl\_dyn\_parametro!pasta\_AutoHotkey

wl\_pasta\_integracao\_gid = wl\_dyn\_parametro!pasta\_integracao\_gid

Set wl\_tbl\_gid\_integracao\_ocorrencia = wl\_dbf.OpenRecordset("tbl\_gid\_integracao\_ocorrencia", dbOpenTable)

wl\_tbl\_gid\_integracao\_ocorrencia.Index = "idx\_valida\_entrada"

wl\_pasta\_script = wg\_sis\_arq\_imp

wl\_arq = "gid.txt"

Set wl\_wsh\_shell = CreateObject("WScript.Shell")

Set wl\_wsh\_fso = CreateObject("Scripting.FileSystemObject")

Set wl\_arq\_script = wl\_wsh\_fso.createtextFile(wl\_pasta\_script & wl\_arq)

wl\_string\_cab = "WinWait, GIDHab [CHC00000]," & vbCrLf & \_

"IfWinNotActive, GIDHab [CHC00000], , WinActivate, GIDHab [CHC00000]," & vbCrLf & \_

"WinWaitActive, GIDHab [CHC00000]," & vbCrLf & \_

"MouseClick , Left, 505, 32" & vbCrLf & \_

"Sleep , " & wl\_tempo\_comando & vbCrLf & \_

"MouseClick , Left, 543, 94" & vbCrLf & \_

"Sleep , " & wl\_tempo\_comando

DoCmd.SetWarnings False

DoCmd.RunSQL "delete \* from tbl\_gid\_integracao\_ocorrencia where id\_gid\_integracao=" & wl\_cod\_gid\_integracao

DoCmd.SetWarnings True

Me.sub\_gid\_integracao\_ocorrencia.Requery

wl\_cont\_instr = 0

Do Until wl\_dyn\_func.EOF

wl\_cont\_instr = wl\_cont\_instr + 1

wl\_razao\_func = wl\_dyn\_func!razao\_func

wl\_rg\_func = bas\_magnetico\_conversao\_caracteres(wl\_dyn\_func!inscr\_icmsrg, "nro")

wl\_arq = wl\_rg\_func & ".txt"

Me.status = "Gerando dados instrutor " & wl\_razao\_func

Me.Repaint

wl\_tbl\_gid\_integracao\_ocorrencia.Seek "=", wl\_cod\_gid\_integracao, wl\_dyn\_func!cod\_func

If wl\_tbl\_gid\_integracao\_ocorrencia.NoMatch = True Then

wl\_tbl\_gid\_integracao\_ocorrencia.AddNew

wl\_tbl\_gid\_integracao\_ocorrencia!id\_gid\_integracao = wl\_cod\_gid\_integracao

wl\_tbl\_gid\_integracao\_ocorrencia!id\_func = wl\_dyn\_func!cod\_func

wl\_tbl\_gid\_integracao\_ocorrencia!inscr\_icmsrg = wl\_dyn\_func!inscr\_icmsrg

wl\_tbl\_gid\_integracao\_ocorrencia.Update

End If

Me.sub\_gid\_integracao\_ocorrencia.Requery

wl\_string = ";" & vbCrLf & \_

"; instrutor " & wl\_razao\_func & vbCrLf & \_

";"

wl\_arq\_script.WriteLine wl\_string\_cab

wl\_arq\_script.WriteLine wl\_string

wl\_arq\_script.WriteLine "WinWait, Relação de Aulas,"

wl\_arq\_script.WriteLine "IfWinNotActive, Relação de Aulas, , WinActivate, Relação de Aulas,"

wl\_arq\_script.WriteLine "WinWaitActive, Relação de Aulas,"

wl\_arq\_script.WriteLine "MouseClick , Left, 151, 58"

wl\_arq\_script.WriteLine "MouseClick , Left, 151, 58"

wl\_arq\_script.WriteLine "Sleep, " & wl\_tempo\_comando

wl\_arq\_script.WriteLine "Send, {DEL}" & wl\_rg\_func & "{TAB}rs{TAB}" & Format(Me.dt\_ini, "ddmmyyyy") & "{TAB}" & Format(Me.dt\_fim, "ddmmyyyy") & "{ALTDOWN}l{ALTUP}"

wl\_arq\_script.WriteLine "Sleep, " & wl\_tempo\_programa

wl\_arq\_script.WriteLine "MouseClick , Right, 373, 211"

wl\_arq\_script.WriteLine "Sleep, " & wl\_tempo\_programa

wl\_arq\_script.WriteLine "Send, {DOWN}{DOWN}{ENTER}"

wl\_arq\_script.WriteLine "Sleep, " & wl\_tempo\_programa

wl\_arq\_script.WriteLine "Send, {LWINDOWN}r{LWINUP}"

wl\_arq\_script.WriteLine "Sleep, " & wl\_tempo\_comando

wl\_arq\_script.WriteLine "WinWait, Executar,"

wl\_arq\_script.WriteLine "IfWinNotActive, Executar, , WinActivate, Executar,"

wl\_arq\_script.WriteLine "WinWaitActive, Executar,"

wl\_arq\_script.WriteLine "Send, notepad{ENTER}"

wl\_arq\_script.WriteLine "Sleep, " & wl\_tempo\_programa

wl\_arq\_script.WriteLine "WinWait, Sem título - Bloco de notas,"

wl\_arq\_script.WriteLine "IfWinNotActive, Sem título - Bloco de notas, , WinActivate, Sem título - Bloco de notas,"

wl\_arq\_script.WriteLine "WinWaitActive, Sem título - Bloco de notas,"

wl\_arq\_script.WriteLine "Send, {ALTDOWN}{ALTUP}el{ALTDOWN}{ALTUP}av"

wl\_arq\_script.WriteLine "WinWait, Salvar como,"

wl\_arq\_script.WriteLine "IfWinNotActive, Salvar como, , WinActivate, Salvar como,"

wl\_arq\_script.WriteLine "WinWaitActive, Salvar como,"

wl\_arq\_script.WriteLine "Send, " & wl\_pasta\_integracao\_gid & wl\_rg\_func & "{ALTDOWN}l{ALTUP}"

wl\_arq\_script.WriteLine "WinWait, " & wl\_arq & " - Bloco de notas,"

wl\_arq\_script.WriteLine "IfWinNotActive, " & wl\_arq & " - Bloco de notas, , WinActivate, " & wl\_arq & " - Bloco de notas,"

wl\_arq\_script.WriteLine "WinWaitActive, " & wl\_arq & " - Bloco de notas,"

wl\_arq\_script.WriteLine "Sleep , " & wl\_tempo\_programa

wl\_arq\_script.WriteLine "Send, {ALTDOWN}{F4}{ALTUP}"

wl\_arq\_script.WriteLine "Sleep, " & wl\_tempo\_programa

wl\_arq\_script.WriteLine "WinWait, Relação de Aulas,"

wl\_arq\_script.WriteLine "IfWinNotActive, Relação de Aulas, , WinActivate, Relação de Aulas,"

wl\_arq\_script.WriteLine "WinWaitActive, Relação de Aulas,"

wl\_arq\_script.WriteLine "Send, {ALTDOWN}{F4}{ALTUP}"

wl\_arq\_script.WriteLine "Sleep, " & wl\_tempo\_programa

wl\_dyn\_func.MoveNext

Loop

wl\_arq\_script.WriteLine "MsgBox,operação concluida. Pressione o botão IMPORTAR ARQUIVOS GID no sistema CFC"

wl\_arq\_script.Close

MsgBox "Verifique se o sistema GID está aberto." & vbCrLf & vbCrLf & "Pressione o botão OK para continuar.", vbInformation, "Atenção !!!"

wl\_string = wl\_pasta\_autohotkey & "AutoHotkey.exe " & wl\_pasta\_script & "gid.txt"

wl\_wsh\_shell.Run wl\_string

Me.status = ""

Me.status.Requery

Exit\_btn\_executar\_script\_Click:

Exit Sub

Err\_btn\_executar\_script\_Click:

If Err > 0 Then

MsgBox "Ocorreu o erro:" & vbCrLf & vbCrLf & Err & " - " & Error, vbInformation, "Atenção !!!"

Resume Exit\_btn\_executar\_script\_Click

End If

End Sub

## Código fonte para executar a crítica dos dados exportados do GID

Private Sub btn\_imp\_arq\_gid\_Click()

On Error GoTo Err\_btn\_imp\_arq\_gid\_Click

Dim wl\_dbf As Database, \_

wl\_dyn As Recordset, \_

wl\_dyn\_gid\_integracao\_ocorrencia As Recordset, \_

wl\_dyn\_parametro As Recordset, \_

wl\_tbl\_pratica\_gid As Recordset, \_

wl\_dyn\_gid\_integracao\_marcacao As Recordset, \_

wl\_dyn\_marcacao As Recordset, \_

wl\_dyn\_cfc As Recordset, \_

wl\_dyn\_gid\_integracao\_registro As Recordset, \_

wl\_wsh\_shell As Object, \_

wl\_wsh\_fso As Object, \_

wl\_wsh\_fso\_arq As Object, \_

wl\_pasta\_integracao\_gid, wl\_arq, wl\_arq\_contem, wl\_arq\_contem\_linhas, \_

wl\_sql, \_

wl\_cod\_func, wl\_rg\_func, wl\_cod\_gid\_integracao, i, wl\_cont\_reg, \_

wl\_dt\_ini As Date, wl\_dt\_fim As Date, wl\_pos\_renach, wl\_pos\_aluno, wl\_pos\_dt, wl\_pos\_h, wl\_pos\_categ, wl\_pos\_quant, wl\_pos\_veiculo, wl\_pos\_registro, wl\_pos\_vlr, \_

wl\_achou, wl\_quant\_cfc

If IsNull(Me.dt\_ini) Or IsNull(Me.dt\_fim) Then

MsgBox "É necessário informar a data inicial e final.", vbInformation, "Atenção !!!"

Exit Sub

End If

wl\_cod\_gid\_integracao = Me.cod\_gid\_integracao

If bas\_renomear\_arq\_gid() = True Then

End If

Set wl\_dbf = OpenDatabase(wg\_sis\_arq\_dbf)

Set wl\_dyn\_parametro = CurrentDb.OpenRecordset("tbl\_parametros", dbOpenDynaset)

wl\_pasta\_integracao\_gid = wl\_dyn\_parametro!pasta\_integracao\_gid

Set wl\_tbl\_pratica\_gid = wl\_dbf.OpenRecordset("tbl\_pratica\_gid", dbOpenTable)

wl\_tbl\_pratica\_gid.Index = "idx\_valida\_entrada"

Set wl\_dyn\_gid\_integracao\_registro = wl\_dbf.OpenRecordset("tbl\_gid\_integracao\_registro", dbOpenTable) 'tabela com dados do arquivo GID

Set wl\_dyn\_gid\_integracao\_marcacao = wl\_dbf.OpenRecordset("tbl\_gid\_integracao\_marcacao", dbOpenTable)

wl\_dyn\_gid\_integracao\_marcacao.Index = "validacao\_entrada"

DoCmd.SetWarnings False

DoCmd.RunSQL "DELETE \* FROM tbl\_gid\_integracao\_registro WHERE id\_gid\_integracao=" & wl\_cod\_gid\_integracao

DoCmd.RunSQL "DELETE \* FROM tbl\_gid\_integracao\_marcacao WHERE id\_gid\_integracao=" & wl\_cod\_gid\_integracao

Set wl\_wsh\_shell = CreateObject("WScript.Shell")

Set wl\_wsh\_fso = CreateObject("Scripting.FileSystemObject")

Set wl\_dyn = Me.sub\_gid\_integracao\_ocorrencia.Form.RecordsetClone

wl\_dyn.MoveFirst

Do Until wl\_dyn.EOF

Me.status = "Verificando criação arquivo instrutor " & wl\_dyn!razao\_func

Me.Repaint

wl\_cod\_func = wl\_dyn!id\_func

'--> Exclui lançamentos importados anteriormente pois pode ter havido exclusão de aulas no GID

DoCmd.SetWarnings False

DoCmd.RunSQL "delete \* from tbl\_pratica\_gid where id\_instrutor = " & wl\_cod\_func & " and dt\_pratica\_gid >=#" & Format(Me.dt\_ini, "mm/dd/yy") & "# and dt\_pratica\_gid <= #" & Format(Me.dt\_fim, "mm/dd/yy") & "#"

' Debug.Print "delete \* from tbl\_pratica\_gid where id\_instrutor = " & wl\_cod\_func & " and dt\_pratica\_gid >=#" & Format(Me.dt\_ini, "mm/dd/yy") & "# and dt\_pratica\_gid <= #" & Format(Me.dt\_fim, "mm/dd/yy") & "#"

DoCmd.SetWarnings True

'Exit Sub

wl\_rg\_func = bas\_magnetico\_conversao\_caracteres(wl\_dyn!inscr\_icmsrg, "nro")

wl\_arq = wl\_rg\_func & ".txt"

wl\_dyn.Edit

If wl\_wsh\_fso.FileExists(wl\_pasta\_integracao\_gid & wl\_arq) Then

wl\_dyn!flag\_arq\_gerado = True

Set wl\_wsh\_fso\_arq = wl\_wsh\_fso.opentextfile(wl\_pasta\_integracao\_gid & wl\_arq, 1)

wl\_arq\_contem = wl\_wsh\_fso\_arq.readall

wl\_arq\_contem\_linhas = Split(wl\_arq\_contem, vbCrLf)

wl\_dt\_ini = Format(Mid(wl\_arq\_contem\_linhas(0), 24, 2) & "/" & Mid(wl\_arq\_contem\_linhas(0), 27, 2) & "/" & Mid(wl\_arq\_contem\_linhas(0), 30, 4), "DD/MM/yy")

wl\_dt\_fim = Format(Mid(wl\_arq\_contem\_linhas(0), 37, 2) & "/" & Mid(wl\_arq\_contem\_linhas(0), 40, 2) & "/" & Mid(wl\_arq\_contem\_linhas(0), 43, 4), "DD/MM/yy")

wl\_pos\_renach = InStr(1, wl\_arq\_contem\_linhas(2), "RENACH")

wl\_pos\_aluno = InStr(1, wl\_arq\_contem\_linhas(2), "Nome")

wl\_pos\_dt = InStr(1, wl\_arq\_contem\_linhas(2), "Data Iníci")

wl\_pos\_h = InStr(1, wl\_arq\_contem\_linhas(2), "Horário")

wl\_pos\_categ = InStr(1, wl\_arq\_contem\_linhas(2), "C ")

wl\_pos\_quant = InStr(1, wl\_arq\_contem\_linhas(2), "N ")

wl\_pos\_veiculo = InStr(1, wl\_arq\_contem\_linhas(2), "Placa")

wl\_pos\_registro = InStr(1, wl\_arq\_contem\_linhas(2), "Data/hora Registro")

wl\_pos\_vlr = InStr(1, wl\_arq\_contem\_linhas(2), "Valor")

wl\_cont\_reg = 0

If Me.dt\_ini <> wl\_dt\_ini Or Me.dt\_fim <> wl\_dt\_fim Then

wl\_dyn!flag\_arq\_validado = False

Else

wl\_dyn!flag\_arq\_validado = True

For i = 3 To UBound(wl\_arq\_contem\_linhas)

If wl\_arq\_contem\_linhas(i) <> "" Then

' Tabelas utilizadas para realizar a auditoria

wl\_dyn\_gid\_integracao\_marcacao.Seek "=", wl\_cod\_gid\_integracao, Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_aluno, wl\_pos\_dt - wl\_pos\_aluno), wl\_rg\_func, Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_dt, 10), "DD/MM/yy"), Trim(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_categ, wl\_pos\_quant - wl\_pos\_categ)), "GID"

If wl\_dyn\_gid\_integracao\_marcacao.NoMatch Then

wl\_dyn\_gid\_integracao\_marcacao.AddNew

wl\_dyn\_gid\_integracao\_marcacao!id\_gid\_integracao = wl\_cod\_gid\_integracao

wl\_dyn\_gid\_integracao\_marcacao!id\_func = wl\_cod\_func

wl\_dyn\_gid\_integracao\_marcacao!rg\_instr = wl\_rg\_func

wl\_dyn\_gid\_integracao\_marcacao!dt\_pratica\_gid = Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_dt, 10), "DD/MM/yy")

wl\_dyn\_gid\_integracao\_marcacao!razao\_aluno\_gid = Trim(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_aluno, wl\_pos\_dt - wl\_pos\_aluno))

wl\_dyn\_gid\_integracao\_marcacao!quant\_gid = Trim(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_quant, 2))

wl\_dyn\_gid\_integracao\_marcacao!id\_categ\_gid = Trim(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_categ, wl\_pos\_quant - wl\_pos\_categ))

wl\_dyn\_gid\_integracao\_marcacao!origem = "GID"

wl\_dyn\_gid\_integracao\_marcacao.Update

Else

wl\_dyn\_gid\_integracao\_marcacao.Edit

wl\_dyn\_gid\_integracao\_marcacao!quant\_gid = wl\_dyn\_gid\_integracao\_marcacao!quant\_gid + Val(Trim(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_quant, 2)))

wl\_dyn\_gid\_integracao\_marcacao.Update

End If

wl\_dyn\_gid\_integracao\_registro.AddNew

wl\_dyn\_gid\_integracao\_registro!id\_gid\_integracao = wl\_cod\_gid\_integracao

wl\_dyn\_gid\_integracao\_registro!id\_func = wl\_cod\_func

wl\_dyn\_gid\_integracao\_registro!dt\_pratica\_gid = Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_dt, 10), "DD/MM/yy")

wl\_dyn\_gid\_integracao\_registro!hr\_pratica\_gid\_ini = Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_h, 5), "hh:mm")

wl\_dyn\_gid\_integracao\_registro!hr\_pratica\_gid\_fim = Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_h + 8, 5), "hh:mm")

wl\_dyn\_gid\_integracao\_registro!dt\_registro\_gid = Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_registro, 10), "DD/MM/yy")

wl\_dyn\_gid\_integracao\_registro!hr\_registro\_gid = Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_registro + 11, 5), "hh:mm")

wl\_dyn\_gid\_integracao\_registro.Update

' Tabela com dados importados do gid

wl\_tbl\_pratica\_gid.Seek "=", wl\_cod\_func, Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_dt, 10), "DD/MM/yy"), Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_h, 5), "hh:mm")

wl\_cont\_reg = wl\_cont\_reg + 1

If wl\_tbl\_pratica\_gid.NoMatch = True Then

wl\_tbl\_pratica\_gid.AddNew

Else

wl\_tbl\_pratica\_gid.Edit

End If

wl\_tbl\_pratica\_gid!id\_instrutor = wl\_cod\_func

wl\_tbl\_pratica\_gid!id\_veiculo = Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_veiculo, 3) & "-" & Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_veiculo + 3, 4)

wl\_tbl\_pratica\_gid!renach = Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_renach, 11)

wl\_tbl\_pratica\_gid!dt\_pratica\_gid = Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_dt, 10), "DD/MM/yy")

wl\_tbl\_pratica\_gid!hr\_pratica\_gid\_ini = Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_h, 5), "hh:mm")

wl\_tbl\_pratica\_gid!hr\_pratica\_gid\_fim = Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_h + 8, 5), "hh:mm")

wl\_tbl\_pratica\_gid!dt\_registro\_gid = Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_registro, 10), "DD/MM/yy")

wl\_tbl\_pratica\_gid!hr\_registro\_gid = Format(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_registro + 11, 5), "hh:mm")

wl\_tbl\_pratica\_gid!qtd\_aula = Trim(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_quant, 2))

wl\_tbl\_pratica\_gid!razao\_aluno = Trim(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_aluno, wl\_pos\_dt - wl\_pos\_aluno))

wl\_tbl\_pratica\_gid!vlr\_total = Trim(Mid(wl\_arq\_contem\_linhas(i), wl\_pos\_vlr, wl\_pos\_veiculo - wl\_pos\_vlr))

wl\_tbl\_pratica\_gid.Update

End If

Next i

End If

'ROTINAS para auditoria

wl\_dt\_ini = Me.dt\_ini

wl\_dt\_fim = Me.dt\_fim

wl\_sql = "SELECT tbl\_func.cod\_func, tbl\_func.inscr\_icmsrg, tbl\_pratica.dt\_pratica, tbl\_aluno.razao\_aluno, Count(tbl\_pratica.cod\_pratica) AS quant, tbl\_pratica.id\_categoria " & \_

"FROM tbl\_aluno INNER JOIN (tbl\_func INNER JOIN tbl\_pratica ON tbl\_func.cod\_func = tbl\_pratica.id\_instrutor) ON tbl\_aluno.cod\_aluno = tbl\_pratica.id\_aluno " & \_

"WHERE dt\_pratica >=#" & Format(wl\_dt\_ini, "mm/dd/yy") & "# and dt\_pratica <=#" & Format(wl\_dt\_fim, "mm/dd/yy") & "# and tbl\_func.inscr\_icmsrg='" & wl\_rg\_func & "' " & \_

"GROUP BY tbl\_func.cod\_func, tbl\_func.inscr\_icmsrg, tbl\_pratica.dt\_pratica, tbl\_aluno.razao\_aluno, tbl\_pratica.id\_categoria " & \_

"ORDER BY tbl\_func.cod\_func, tbl\_func.inscr\_icmsrg, tbl\_pratica.dt\_pratica, tbl\_aluno.razao\_aluno;"

Debug.Print 1, wl\_sql

Set wl\_dyn\_cfc = CurrentDb.OpenRecordset(wl\_sql, dbOpenDynaset)

Do Until wl\_dyn\_cfc.EOF

wl\_dyn\_gid\_integracao\_marcacao.Seek "=", wl\_cod\_gid\_integracao, bas\_cpo\_limpar\_especial(wl\_dyn\_cfc!razao\_aluno), wl\_dyn\_cfc!inscr\_icmsrg, wl\_dyn\_cfc!dt\_pratica, wl\_dyn\_cfc!id\_categoria, "GID"

If wl\_dyn\_gid\_integracao\_marcacao.NoMatch Then

wl\_dyn\_gid\_integracao\_marcacao.AddNew

wl\_dyn\_gid\_integracao\_marcacao!id\_gid\_integracao = wl\_cod\_gid\_integracao

wl\_dyn\_gid\_integracao\_marcacao!id\_func = wl\_dyn\_cfc!cod\_func

wl\_dyn\_gid\_integracao\_marcacao!rg\_instr = wl\_dyn\_cfc!inscr\_icmsrg

wl\_dyn\_gid\_integracao\_marcacao!dt\_pratica\_gid = wl\_dyn\_cfc!dt\_pratica

wl\_dyn\_gid\_integracao\_marcacao!razao\_aluno\_gid = bas\_cpo\_limpar\_especial(wl\_dyn\_cfc!razao\_aluno)

wl\_dyn\_gid\_integracao\_marcacao!quant\_gid = wl\_dyn\_cfc!quant

wl\_dyn\_gid\_integracao\_marcacao!id\_categ\_gid = wl\_dyn\_cfc!id\_categoria

wl\_dyn\_gid\_integracao\_marcacao!origem = "CFC"

wl\_dyn\_gid\_integracao\_marcacao!id\_status = "2"

wl\_dyn\_gid\_integracao\_marcacao.Update

Else

wl\_dyn\_gid\_integracao\_marcacao.Edit

wl\_dyn\_gid\_integracao\_marcacao!id\_status = "1"

If wl\_dyn\_gid\_integracao\_marcacao!quant\_gid <> wl\_dyn\_cfc!quant Then

wl\_dyn\_gid\_integracao\_marcacao!flag\_erro\_quant = -1

Else

wl\_dyn\_gid\_integracao\_marcacao!flag\_erro\_quant = 0

End If

wl\_dyn\_gid\_integracao\_marcacao!quant\_cfc = wl\_dyn\_cfc!quant

wl\_dyn\_gid\_integracao\_marcacao.Update

End If

wl\_dyn\_cfc.MoveNext

Loop

wl\_sql = "SELECT \* FROM tbl\_gid\_integracao\_marcacao WHERE id\_gid\_integracao = " & cod\_gid\_integracao & " and rg\_instr = '" & wl\_rg\_func & "' and origem='GID'"

Debug.Print wl\_sql

Set wl\_dyn\_marcacao = CurrentDb.OpenRecordset(wl\_sql, dbOpenDynaset)

wl\_achou = False

Do Until wl\_dyn\_marcacao.EOF

If wl\_dyn\_cfc.RecordCount > 0 Then wl\_dyn\_cfc.MoveFirst

wl\_dyn\_cfc.FindFirst "razao\_aluno='" & wl\_dyn\_marcacao!razao\_aluno\_gid & "'"

wl\_achou = False

Do Until wl\_dyn\_cfc.EOF

If wl\_dyn\_marcacao!razao\_aluno\_gid = bas\_cpo\_limpar\_especial(wl\_dyn\_cfc!razao\_aluno) And wl\_dyn\_marcacao!rg\_instr = wl\_dyn\_cfc!inscr\_icmsrg And wl\_dyn\_marcacao!dt\_pratica\_gid = wl\_dyn\_cfc!dt\_pratica And wl\_dyn\_marcacao!id\_categ\_gid = wl\_dyn\_cfc!id\_categoria And wl\_dyn\_marcacao!origem = "GID" Then

wl\_quant\_cfc = wl\_dyn\_cfc!quant

wl\_achou = True

Exit Do

End If

wl\_dyn\_cfc.MoveNext

Loop

wl\_dyn\_marcacao.Edit

If wl\_achou Then

wl\_dyn\_marcacao!id\_status = "1"

If wl\_dyn\_marcacao!quant\_gid <> wl\_quant\_cfc Then

wl\_dyn\_marcacao!flag\_erro\_quant = -1

Else

wl\_dyn\_marcacao!flag\_erro\_quant = 0

End If

wl\_dyn\_marcacao!quant\_cfc = wl\_quant\_cfc

Else

wl\_dyn\_marcacao!id\_status = "3"

End If

wl\_dyn\_marcacao.Update

wl\_dyn\_marcacao.MoveNext

Loop

wl\_dyn!qtd\_lancto = wl\_cont\_reg 'UBound(wl\_arq\_contem\_linhas) - 3

wl\_wsh\_fso\_arq.Close

wl\_wsh\_fso.Copyfile wl\_pasta\_integracao\_gid & wl\_arq, wg\_sis\_arq\_bkp & wl\_arq, True

wl\_wsh\_fso.Deletefile wl\_pasta\_integracao\_gid & wl\_arq

Else

wl\_dyn!flag\_arq\_gerado = False

End If

wl\_dyn.Update

wl\_dyn.MoveNext

Loop

Call bas\_gid\_verifica\_registro

Me.status = ""

Me.status.Requery

MsgBox "Operação concluída.", vbInformation, "Atenção !!!"

DoCmd.OpenReport "rpt\_gid\_integracao\_a1", acViewPreview, , "id\_gid\_integracao = " & Me.cod\_gid\_integracao

Wscript.Quit

Exit\_btn\_imp\_arq\_gid\_Click:

Exit Sub

Err\_btn\_imp\_arq\_gid\_Click:

If Err > 0 Then

MsgBox "Ocorreu o erro:" & vbCrLf & vbCrLf & Err & " - " & Error, vbInformation, "Atenção !!!"

Resume Exit\_btn\_imp\_arq\_gid\_Click

End If

End Sub

## Código fonte para verificar conflitos entre os horários

O código fonte abaixo tem como função, verificar a existência de conflitos entre os horários em que foram ministradas aulas com o horário em que foi efetuado o lançamento no sistema GID.

Private Function bas\_gid\_verifica\_registro()

Dim wl\_dbf As Database, \_

wl\_dyn\_registro As Recordset, \_

wl\_dyn\_registro\_tmp As Recordset, \_

wl\_sql

Set wl\_dbf = OpenDatabase(wg\_sis\_arq\_dbf)

wl\_sql = "select \* from tbl\_gid\_integracao\_registro where id\_gid\_integracao=" & Me.cod\_gid\_integracao & " order by dt\_registro\_gid"

Set wl\_dyn\_registro = CurrentDb.OpenRecordset(wl\_sql, dbOpenDynaset)

With wl\_dyn\_registro

Do Until .EOF

wl\_sql = "select \* from tbl\_gid\_integracao\_registro where id\_gid\_integracao=" & Me.cod\_gid\_integracao & " and dt\_pratica\_gid=#" & Format(!dt\_registro\_gid, "mm/dd/yy") & "#" ' and cod\_gid\_registro <> " & !cod\_gid\_registro

Set wl\_dyn\_registro\_tmp = CurrentDb.OpenRecordset(wl\_sql, dbOpenDynaset)

Do Until wl\_dyn\_registro\_tmp.EOF

' Debug.Print !dt\_registro\_gid, !hr\_registro\_gid, wl\_dyn\_registro\_tmp!hr\_pratica\_gid\_ini, wl\_dyn\_registro\_tmp!hr\_pratica\_gid\_fim

If !hr\_registro\_gid >= wl\_dyn\_registro\_tmp!hr\_pratica\_gid\_ini And !hr\_registro\_gid <= wl\_dyn\_registro\_tmp!hr\_pratica\_gid\_fim Then

' MsgBox "conflito" & vbCrLf & wl\_dyn\_registro\_tmp!dt\_pratica\_gid & " " & !dt\_registro\_gid & vbCrLf & wl\_dyn\_registro\_tmp!hr\_registro\_gid & vbCrLf & !hr\_pratica\_gid\_ini & " " & !hr\_pratica\_gid\_fim

.Edit

!flag\_erro = True

!obs = !obs & "Conflito c/ data: " & !dt\_registro\_gid & " hora aula: " & wl\_dyn\_registro\_tmp!hr\_pratica\_gid\_ini & " " & wl\_dyn\_registro\_tmp!hr\_pratica\_gid\_fim & " - hora registro: " & !hr\_registro\_gid

.Update

End If

wl\_dyn\_registro\_tmp.MoveNext

Loop

.MoveNext

Loop

End With

End Function

## Procedimento operacional para integração entre os sistemas CFC e GID

A integração entre os sistemas é realizada através da captura de dados do GID pelo sistema CFC

É muito importante seguir as etapas na ordem.

Caso ocorra qualquer erro, os sistemas devem ser finalizados e o processo deve ser iniciado novamente.

|  |  |  |
| --- | --- | --- |
| **Sistema** | **Etapa** | **Informações de Apoio** |
| **Sistema  GID** | Abrir o sistema GID | * + - Informar o usuário e senha (necessário ser diretor)     - Fechar todas as janelas que estiverem abertas |
| Minimizar o sistema GID |  |
| **Sistema  CFC** | Abrir o sistema CFC | * + - Informar o usuário e senha (necessário ser diretor) |
| Acessar a auditoria | * + - Clicar no menu Sistema (Item A)     - Selecionar a opção “Crítica GID Integrada” (item B)     **B**  **A** |
| Executar *Script* de Captura  **D**  **A** | * + - Informar o Período, preenchendo os campos data inicial e data final, ao qual deseja obter os dados do sistema GID(Item A)     - Clicar no botão “Executar *Script* Captura dados Sistema GID” (item B)   **E**  **C**  **B** |
| **AutoHotkey** | Aguardar | * + - Aguardar a execução do *script* até ser apresentada a mensagem “Operação concluída. Pressione o botão IMPORTAR ARQUIVOS GID no sistema CFC |
| **Sistema  CFC** | Importar os dados | * + - Clicar sobre o botão “Importar arquivos GID” (Item C) |
| Verificar as Inconsistências | * + - No item D é apresentado o nome dos instrutores, a quantidade de lançamentos e de Inconsistências encontradas |
| Imprimir o relatório | * + - Clicar no botão “Imprimir relatório de Auditoria” (item E)   O relatório apresenta o resumo das inconsistências por instrutor |

1. Programa para execução de *scripts* através da linha de comando [↑](#footnote-ref-1)
2. *Component Object Model* (COM) é uma plataforma da Microsoft para componentes de software lançada em 1993. Ela é usada para permitir a comunicação entre processos e a criação dinâmica de objetos em qualquer linguagem de programação que suporte a tecnologia. [↑](#footnote-ref-2)
3. linguagem de *script* para automação de tarefas [↑](#footnote-ref-3)
4. Programa que permite transmitir e receber sinais de controle remoto infravermelho para controlar o computador site: http://winlirc.sourceforge.net/ [↑](#footnote-ref-4)
5. Linguagem de *script* projetado para automatizar tarefas no Windows e criar *scripts* em geral. Site: http://www.autoitscript.com [↑](#footnote-ref-5)
6. O roteiro esta em apêndice sob o título “Procedimento operacional para integração entre os sistemas CFC e GID” [↑](#footnote-ref-6)