UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
ESCOLA DE INFORMÁTICA APLICADA

O Uso do AutoHotkey como Ferramenta de Automação em Ambiente Windows

Antonio França da Guia

Orientadora
Morganna Carmem Diniz

RIO DE JANEIRO, RJ – BRASIL
DEZEMBRO DE 2015

O Uso do AutoHotkey como Ferramenta de Automação em Ambiente Windows

Antonio França da Guia

Projeto de Graduação apresentado à Escola de Informática Aplicada da Universidade Federal do Estado do Rio de Janeiro (UNIRIO) para obtenção do título de Bacharel em Sistemas de Informação.

Aprovada por:

_____

Prof.ª Morganna Carmem Diniz, D.Sc. (UNIRIO)

_____

Prof.ª Flávia Maria Santoro, D.Sc. (UNIRIO)

_____

Prof. Sidney Cunha de Lucena, D.Sc. (UNIRIO)

RIO DE JANEIRO, RJ – BRASIL.
DEZEMBRO DE 2015

Agradecimentos

A minha família, pelos motivos mais óbvios.

A minha namorada, pelo amor imensurável e apoio incondicional.

A todos os velhos e novos amigos, dentro e fora da faculdade, por tornarem esta jornada uma experiência divertida e prazerosa, cheia de aprendizado e boas lembranças.

À professora Morganna, pela atenção, objetividade, confiança e incentivo, permitindo que fosse abordado neste trabalho um tema que eu desejava há muito tempo.

Aos professores Flávia e Sidney, pelas observações, elogios, interesse em participar da banca e curiosidade pelo assunto.

Aos professores do curso, pela atenção, dedicação e vontade de lecionar, acreditando nos alunos e proporcionando a estes um ambiente de aprendizado receptivo e amigável, como toda universidade deveria ser.

Aos funcionários da secretaria do curso, pela paciência, dedicação e alegria, provando definitivamente que uma secretaria universitária pode trabalhar de forma rápida, eficaz e atenciosa, atendendo as pessoas sempre com um sorriso no rosto.

RESUMO

Apesar das recentes preocupações ao longo das últimas décadas, muitos softwares não estão otimizados atualmente para acessibilidade e usabilidade. Enquanto isso, a automação de tarefas tem se tornado cada vez mais popular entre os chamados “usuários finais”, que tipicamente não estão familiarizados com programação e desenvolvimento de software. Consequentemente, vem ocorrendo um aumento no interesse por ferramentas que permitam que tais usuários personalizem as interações com seus computadores – tanto em residências como em locais de trabalho. Este artigo busca explorar a linguagem de script AutoHotkey em seu contexto completo através do levantamento de diversas literaturas e recursos relacionados, bem como o desenvolvimento de uma solução prática para a automação de uma tarefa acadêmica específica. Esperamos que esta pesquisa sirva como referências para trabalhos futuros.

Palavras-chave: AutoHotkey, AHK, Automação, Windows, Software, Acessibilidade.

ABSTRACT

Despite recent concerns throughout the last decades, many softwares are currently still not optimized for accessibility and usability. Meanwhile, task automation has become increasingly popular among the so called “final users”, typically not familiar with programming and software development. Consequently, there has been a spike of interest in tools which allow users themselves to customize the interactions with their computers – both in residences and workplaces. This article attempts to explore the AutoHotkey scripting language in its full context by surveying various related literatures and resources, as well as developing a practical solution for automating a specific academic task. We hope this research may serve as a reference for future works.

Keywords: AutoHotkey, AHK, Automation, Windows, Software, Accessibility.

Índice

1 Introdução 9
1.1 Motivação 10
1.2 Objetivos 11
1.3 Organização do texto 12
2 Conceitos 13
2.1 AutoHotkey 13
2.1.1 História 13
2.1.1.1 Origem 13
2.1.1.2 Transição 14
2.1.1.3 Atualidade 15
2.1.2 Especificações Técnicas 15
2.1.2.1 Ferramentas Adicionais e Características Operacionais 15
2.1.2.2 Funcionalidades, Sintaxe e Exemplos 17
2.2 Comparação com Softwares Similares 21
2.2.1 Alternativas 21
2.2.1.1 AutoIt 21
2.2.1.2 SikuliX 23
2.2.2 Breve Análise Comparativa 24
3 Estudo De Caso 26
3.1 Objetivo 26
3.2 Escolha da Atividade 26
3.3 Entendimento do Problema 27
3.4 Desenvolvimento da Solução 32
4 Conclusão 39
4.1 Considerações Finais 39
4.2 Limitações e Dificuldades 40
4.3 Trabalhos Futuros 40
Referências Bibliográficas 41
Apêndice 1: Código Fonte do Script Desenvolvido 53

Índice de Tabelas

Tabela 1: Comparação entre Sikuli, AutoIt e AutoHotkey 25

Índice de Figuras

Figura 1: Página sobre o “AutoIt Script Editor” em *autoitscript.com* 22
Figura 2: Exemplo de visualização de script no “Sikuli IDE” 23
Figura 3: Página em *readthedocs.org* sobre compatibilidade com outras linguagens . 24
Figura 4: Janela principal da “Aplicação de Navegação do SIE” 27
Figura 5: Janela principal da funcionalidade “Matrícula por Aluno” 28
Figura 6: Janela para localização e seleção de aluno / matrícula 28
Figura 7: Janela “Matrícula por Aluno” atualizada com a matrícula selecionada 29
Figura 8: Janela “Matrícula por Aluno” em estado de nova inscrição 30
Figura 9: Janela para localização e seleção de turmas “vazia” 30
Figura 10: Janela para localização e seleção de turmas “preenchida” 31
Figura 11: Janela de “Matrícula por Aluno” atualizada com turma(s) 32
Figura 12: “Active Window Info” – software “espião de janela” 34
Figura 13: Mensagem de erro exibida pelo script 36
Figura 14: Mensagem de sucesso exibida pelo script 38
Figura 15: Arquivos necessários para o uso da solução desenvolvida 38

1 Introdução

Automação é um termo que pode ser definido como a execução feita por uma máquina (como um computador) de uma função que era previamente executada por um humano [1,2]. A automação de tarefas está hoje presente não apenas em diversos ramos da indústria, mas também em nossos lares e até mesmo em nossos bolsos (graças aos *smartphones*), com uma tendência de crescimento contínuo [3]. Ao longo das últimas décadas, incentivada pelos avanços tecnológicos, a sociedade criou máquinas cada vez mais complexas para aprimorar aspectos como produção, conforto e segurança. Algumas funções – especialmente em áreas bem estruturadas e previsíveis de nossas vidas – são comumente aceitas como automatizáveis, onde a interação humana se resume a ligar e desligar uma máquina ou programa, monitorando esporadicamente seu funcionamento [4,5].

Para compreender como um software A pode realizar uma automação que aprimore o uso de um outro software B, é necessário mencionar dois conceitos fortemente relacionados: acessibilidade e usabilidade. Acessibilidade pode ser definida como a possibilidade e condição de qualquer pessoa, independentemente de suas capacidades físico-motoras e perceptivas, culturais e sociais, alcançar os elementos funcionais do ambiente construído, permitindo assim a sua utilização [6]. Estimativas recentes mostram que uma parcela ainda muito pequena dos softwares e websites cumprem efetivamente normas e orientações de acessibilidade [7]. Tecnologias assistivas baseadas em software são tipicamente preferidas por usuários com deficiências, pois ao contrário de acessórios físicos como capacetes, luvas e óculos especiais, apresentam a vantagem de serem menos perceptíveis e consequentemente tendem a chamar menos atenção [8]. Além disso, a acessibilidade é importante tanto para pessoas com deficiências quanto para pessoas que não as possuem, pois as funcionalidades e conceitos aplicados no desenvolvimento de software para pessoas com deficiências

podem ser utilizados por qualquer pessoa para tornar sua interação mais fácil e eficiente [9,10].

Já a usabilidade pode ser interpretada pela combinação de diferentes atributos [11]. Ela é definida pela ISO 9241 como sendo a medida na qual um produto pode ser usado por um determinado perfil de usuário para alcançar objetivos com eficácia, eficiência e satisfação em um contexto específico de uso [12]. Para este trabalho, focaremos especificamente na questão da eficiência, que pode ser percebida, avaliada e comparada de maneira informal mais facilmente. Afirmamos, por exemplo, que a necessidade de executar cliques com o mouse e/ou pressionar muitas teclas do teclado em grandes quantidades e/ou repetitivamente são facilmente percebidas por muitos usuários, sem a necessidade de utilizar uma metodologia formal de avaliação de usabilidade. As consequências negativas desta interação física excessiva para a saúde do usuário já são conhecidos [13] e são tipicamente causadas por falhas no design do software [14].

Finalmente, a despreocupação com acessibilidade e usabilidade em softwares torna-se um problema ainda maior quando o usuário não possui meios para opinar diretamente no desenvolvimento e manutenção de tais sistemas. Essa realidade pode afetar inclusive funcionários de grandes empresas privadas e instituições públicas [15][16]. Com isso, percebe-se um crescimento na adoção de softwares capazes de criar "macros" e "scripts". Vale lembrar que embora estes dois termos possuam origens distintas, são frequentemente confundidos – e até mesmo usados como sinônimos [17] – por apresentarem significados muito parecidos. Em geral, estes termos denotam um conjunto de instruções escritas em alguma linguagem, utilizado justamente para automatizar tarefas através da execução de checagens lógicas, simulações de dispositivos de entrada e outras ações internas do sistema operacional.

Nesse contexto, apresentaremos a linguagem de script AutoHotkey como uma ferramenta robusta para realização de tarefas e automações avançadas.

1.1 Motivação

Uma das motivações para este trabalho está em minha afinidade pessoal com o AutoHotkey, que começou em 2007 quando descobri a existência da linguagem e passei a utilizá-la rotineiramente. Com diversas contribuições e uma presença online ativa,

tornei-me moderador da comunidade oficial em 2012 e desde então tenho buscado oportunidades de trazer um maior reconhecimento para a linguagem.

Apesar da presença do AutoHotkey como facilitador de tarefas em diversos artigos e publicações [18-29] (inclusive em chinês [30-34]) e a existência de alguns livros digitais em inglês sobre a ferramenta [35-38], ainda é difícil encontrar literaturas acadêmicas formais que abordem seu uso cotidiano e apresentem a linguagem de maneira mais ampla – especialmente em português [39,40]. O software chega a ser parte essencial de uma pesquisa realizada por brasileiros [41], mas o potencial e as características da ferramenta não são o foco principal do trabalho.

Por fim, o sistema operacional Windows é ainda hoje o mais utilizado ao redor do mundo [42,43]. Diversos usuários deste sistema executam em sua rotina atividades em aplicações de *desktop* muito específicas, repetitivas e/ou que não podem ser modificadas por algum motivo [44-46]. Este problema abre mais um precedente para a realização deste trabalho, tornando irrelevante a restrição de sistema operacional imposta pelo uso do AutoHotkey.

1.2 Objetivos

Diante da ausência de artigos mais aprofundados, o principal objetivo deste trabalho é apresentar o potencial do AutoHotkey como ferramenta de automação, buscando construir uma literatura acadêmica formal e detalhada sobre a ferramenta. Além de apresentar uma base teórica mais completa para servir como referência para eventuais trabalhos futuros, realizamos também uma aplicação prática do software através de um estudo de caso, onde automatizamos uma tarefa repetitiva de um sistema utilizado na universidade por funcionários da secretaria do curso.

Também estabelecemos como uma outra meta em potencial o ensino da ferramenta aos funcionários da secretaria durante o desenvolvimento do trabalho prático. Definimos que seria interessante se tais participantes acompanhassem de perto o processo de automação. Assim, seria possível observar o nível de interesse deles no AutoHotkey e avaliar suas impressões pessoais sobre a linguagem.

1.3 Organização do texto

O presente trabalho está estruturado em capítulos e, além desta introdução, será desenvolvido da seguinte forma:

- Capítulo II: apresenta a história completa da ferramenta AutoHotkey, realizando especificações técnicas que abrangem desde os programas que acompanham o software até exemplos de suas funcionalidades, bem como uma rápida análise comparativa entre o AutoHotkey e outras alternativas populares de automação.

- Capítulo III: aborda o desenvolvimento do trabalho através de um estudo de caso, detalhando todos os procedimentos efetuados para aplicar na prática uma solução de automação para uma atividade repetitiva em ambiente acadêmico.

- Capítulo IV: reúne as considerações finais, assinala as contribuições da pesquisa e sugere possibilidades de aprofundamento posterior.

2 Conceitos

2.1 AutoHotkey

O AutoHotkey (AHK) é um software gratuito, de código aberto, capaz de criar scripts que automatizam a execução de tarefas repetitivas em ambientes Windows. Mais do que isso, é uma linguagem de script completa que foi concebida inicialmente para facilitar a criação e uso de teclas de atalho, mas que evoluiu com o tempo para se tornar mais completa e permitir o desenvolvimento de scripts mais complexos e até mesmo de aplicações inteiras [47,48].

O próprio website do AutoHotkey indica a possibilidade de alcançar vários objetivos com a linguagem [48]. Exemplos para iniciantes incluem automatizar quase qualquer coisa que envolva o uso de mouse e teclado, criar atalhos e remapear teclas (e botões até mesmo de joysticks), expandir automaticamente abreviações, e converter scripts em arquivos executáveis que podem ser utilizados em computadores que não possuam o AHK instalado. Já os exemplos para usuários mais avançados sugerem a criação de interfaces gráficas personalizadas, bem como a utilização de programação orientada a objetos, funções variádicas, chamadas de DLLs e expressões regulares.

2.1.1 História

2.1.1.1 Origem

No início de 2003, Chris Mallet era um usuário de outro software de automação chamado AutoIt (mencionado na seção 2.2.1.1 deste trabalho). Mallet ficou desapontado com a ausência de um suporte robusto e nativo à criação e utilização de teclas de atalho, e começou em abril a trabalhar em um código próprio que permitisse tais funcionalidades. Ele chegou a propor em setembro uma integração na "versão 2" do

AutoIt, oferecendo inclusive tornar-se o responsável pelo suporte a teclas de atalho, mas não obteve resposta significativa dos desenvolvedores [49].

Assim, a primeira versão beta do AutoHotkey foi lançada na internet em 10 de novembro de 2003 [49,50]. Ainda no fim de 2003, Mallet registrou o domínio "autohotkey.com" para hospedar o portal oficial da comunidade [51]. Em seguida, foi inaugurado também um fórum (cujas primeiras postagens datam de março de 2004 [52]). Em abril de 2005, foi registrado o domínio "autohotkey.net" para hospedar páginas pessoais de membros da comunidade que desejassem armazenar seus trabalhos relacionados ao AutoHotkey [53,54].

2.1.1.2 Transição

O AHK adquiriu mais adeptos e popularidade ao longo do tempo [55-58] e vem sendo utilizado inclusive em muitos trabalhos acadêmicos (como visto na seção 1.1). De acordo com Mallet, uma de suas motivações originais para a concepção do AutoHotkey foi a ideia de empoderar usuários finais – inclusive que não fossem programadores – para que eles mesmos pudessem definir suas próprias teclas de atalhos e rotinas de automação. Seu propósito era criar uma linguagem simples porém flexível, com uma sintaxe acessível e sem que ela se tornasse uma "linguagem de programação completa". Entretanto, após alguns anos, observando que a parcela mais ativa da comunidade desejava funcionalidades de programação mais complexas (como vetores e objetos), Mallet começou a perder o interesse pelo projeto [59].

Com a presença cada vez mais escassa de Mallet e o fim de suas contribuições na versão 1.0.48.05 do software, um usuário proeminente da comunidade chamado Steve Gray – que já havia criado no passado versões personalizadas do AutoHotkey (conhecidas inicialmente como AutoHotkey_L ou AHK_L) – tomou a decisão de dar continuidade ao projeto. Até o último semestre de 2012, diversas das funcionalidades mencionadas anteriormente foram implementadas na linguagem e a comunidade passou a aceitar as novas versões de Gray como oficiais [59]. Também em 2012, Mallet optou por transferir o controle dos servidores do AutoHotkey (que hospedavam todo o conteúdo dos domínios "autohotkey.com" e "autohotkey.net") para um outro usuário antigo na comunidade, chamado Aadil Ahmed [60].

2.1.1.3 Atualidade

No final de 2012, o serviço do domínio "autohotkey.net" foi alvo de hackers. Após encerrar o serviço e fazer diversas alterações no fórum oficial sem aviso prévio, Ahmed tornou-se ausente ao longo de 2013, enfurecendo grande parte dos usuários que utilizavam o fórum diariamente. Após muitos desentendimentos, em outubro daquele ano, o usuário também de longa data chamado Charlie Simmons criou outro portal no domínio "ahkscript.org" – para onde muitos moderadores e membros proeminentes migraram, deixando o fórum oficial em estado de relativo abandono. Em abril de 2014, com a participação de Mallet, a entidade legal "AutoHotkey Foundation LLC" foi criada nos Estados Unidos, a fim principalmente de garantir a continuidade do software e o direito de propriedade a múltiplos representantes, evitando assim a recorrência de problemas passados. No início de 2015, o acesso ao servidores oficiais foi garantido a Simmons e outros membros da entidade. Ahmed teve seus privilégios de administrador revogados e foi iniciado no final do ano um processo de unificação entre os dois portais ("autohotkey.com" e "ahkscript.org"), a ser finalizado no início de 2016 [60].

2.1.2 Especificações Técnicas

2.1.2.1 Ferramentas Adicionais e Características Operacionais

Um fator interessante do AHK é a sua portabilidade. O AutoHotkey é uma linguagem de script (interpretada). Seu interpretador, que converte e executa scripts em tempo real, consiste em um único arquivo executável. Com isso, ao contrário de outras linguagens interpretadas (como Java), scripts criados para AutoHotkey podem ser executados sem a necessidade de nenhum tipo de instalação. Estão disponíveis atualmente três tipos de interpretador diferentes: ANSI 32-bit (para melhor compatibilidade com alguns scripts antigos), Unicode 32-bit (recomendado para novos scripts) e Unicode 64-bit (para melhor desempenho em sistemas de 64 bits) [61,62].

Para aprimorar ainda mais a portabilidade de qualquer aplicação feita usando AHK, existe ainda um compilador. Este compilador na realidade apenas agrupa o interpretador e o próprio script do usuário, gerando um único arquivo executável. Mesmo com todas estas opções, para usuários que não necessitam de tanta portabilidade ou que criam muitos scripts rotineiramente, existe um instalador padrão de AutoHotkey que armazena no disco rígido os componentes necessários e associa arquivos com a extensão ".ahk" ao interpretador [61,63].

Outro tipo de ferramenta incluída em instalações do AutoHotkey desde as primeiras versões (1.0) é um "espião de janela". Este tipo de ferramenta exibe em tempo real informações relevantes sobre uma janela que esteja ativa (ou seja, com a qual o usuário esteja interagindo diretamente em determinado momento). Tais informações são tipicamente úteis para auxiliar na construção manual de scripts, podendo ser usadas como parâmetros para comandos do script ou mesmo para checagens lógicas mais avançadas dependendo do caso de uso em questão. Nas versões mais antigas, era fornecida a ferramenta "Window Spy" (criada na época por Larry Keys para ser utilizada com o AutoIt) [64,65]. Nas versões mais recentes, esta ferramenta foi substituída por outra – escrita inclusive em AutoHotkey [66].

Uma crítica frequente ao AutoHotkey é que, por questões de praticidade, outros softwares úteis para o desenvolvimento de scripts deveriam acompanhar sua instalação. Ainda assim, o novo site da comunidade [48] sugere algumas ferramentas adicionais:

- SciTE4AutoHotkey: ambiente de desenvolvimento integrado (IDE) que consiste em um editor de scripts com várias funcionalidades, como destaque de sintaxe, autocomplemento, *debug* interativo, etc. [67]

- GUI Creator: ferramenta para facilitar a criação de um interface gráfica (GUI) [68]

- Macro Creator: ferramenta completa criada por um brasileiro que permite a geração de scripts de AutoHotkey através da gravação de macros, contando ainda com uma interface intuitiva e outras funcionalidades e opções adicionais [69,70]

- iWB2 Learner ou iWebBrowser2: um coletor de informações sobre páginas da web acessadas através do Internet Explorer – útil para automações do navegador através de objetos COM (Component Object Model) [71]

A Windows API é uma interface presente em sistemas operacionais Windows, responsável por expor serviços e comandos internos do sistema para aplicações. Através desta API, é possível executar ações como exibir interfaces gráficas, acessar recursos do sistema e formatar textos. Embora seja possível utilizar outras linguagens de programação, a Windows API foi desenvolvida com foco em aplicações criadas em C/C++ [72]. Facilitar a interação com esta API é justamente um dos motivos pelo qual o AutoHotkey é uma linguagem de script que foi criada utilizando internamente C++

[73,74]. Essa escolha também traz benefícios inerentes ao C++, como a rapidez e o uso eficiente de memória [75].

Por ter sido idealizado como um software relativamente simples, o AutoHotkey não apresenta a capacidade de uma verdadeira execução multitarefa (do inglês *multithreading*). O software considera uma *thread* como simplesmente um fluxo de execução disparado por algum evento e o fluxo em andamento será sempre aquele disparado mais recentemente. Isso implica na interrupção momentânea de qualquer fluxo anterior, que é retomado ao término do fluxo atual [76]. Apesar dessa característica não representar um problema grave para a maioria dos usuários, alguns optaram por iniciar uma linha de desenvolvimento baseada na versão atual do AutoHotkey (assim como fez o AHK_L em suas primeiras versões). Essa derivação, denominada AutoHotkey_H ou AHK_H, utiliza arquivos DLL e outras técnicas um pouco mais avançadas para adicionar essa funcionalidade na linguagem [77,78].

O AHK é uma linguagem fracamente tipada (como Python e PHP), permitindo que o tipo de dados armazenados em uma variável possa ser alterado dinamicamente durante a execução do programa [79]. Seus paradigmas de programação se assemelham muito aos de C++, permitindo uma programação tanto procedural como orientada a objetos. Assim como programas em C e C++ possuem um método principal chamado "main()", o AutoHotkey inicia a execução de um script sequencialmente a partir de sua primeira linha, até encontrar um comando de retorno ("Return") ou um comando que finalize toda a execução do script, como "Reload" ou "ExitApp" [61].

2.1.2.2 Funcionalidades, Sintaxe e Exemplos

A sintaxe da linguagem é bastante única e divide opiniões. Formulada inicialmente para ser intuitiva (como mencionado na seção 2.1.1.2), ela apresenta características que se assemelham à sintaxe de diversas outras linguagens, como BASIC, Perl, C, Pascal e MS-DOS [80-82]. Apesar de fornecer diversas funções nativas, as principais ações do AutoHotkey são executadas através de comandos. Comandos funcionam como as típicas funções de outras linguagens, mas apresentam seus parâmetros separados por vírgulas e interpretam tais parâmetros (em sua grande maioria) como texto. Segue abaixo um exemplo que processa um texto utilizando uma função, cujo resultado é armazenado em uma variável. Em seguida, o script exibe uma simples caixa de diálogo

na tela, onde o único parâmetro do comando “MsgBox” é interpretado como um texto literal e o trecho “%xyz%” se refere ao conteúdo da variável mencionada anteriormente.

```
xyz := SubStr("Apenas os números vão sobrar: 123456",31)
MsgBox, O conteúdo da variável 'xyz' é: %xyz%
```

Embora este tipo de construção seja relativamente simples, as diferenças existentes entre funções e comandos podem gerar dúvidas para usuários iniciantes. Especificamente, atribuições de valores para variáveis podem ser feitas de múltiplas maneiras, obtendo o mesmo resultado. O próximo exemplo ilustra também uma outra fonte de erros muito comum: o AutoHotkey possibilita a utilização de um sinal de porcentagem no início de um parâmetro para sinalizar que tal parâmetro deve ser interpretado como uma expressão (sendo efetivamente a mesma coisa que um parâmetro de uma função). Assim, textos literais precisam ser declarados entre aspas. Além disso, o exemplo também mostra como textos e conteúdos de variáveis podem ser concatenados em expressões apenas utilizando-os sequencialmente, sem a necessidade de nenhum operador intermediário [63].

```
abc = teste ; atribuição de texto literal
xyz := "teste" ; atribuição de texto em forma de expressão
MsgBox, % "O texto '" abc "' é igual ao texto '" xyz "'."
```

Nesse caso, observamos que a existência de múltiplas formas de alcançar o mesmo objetivo podem ser prejudiciais para a linguagem, pois confundem e dificultam o aprendizado [83]. A própria documentação do AHK chama atenção para este fato, mencionando que o único parâmetro do comando “Sleep” é capaz de processar uma expressão – o que faz com que todas as chamadas do comando no código abaixo produzam o mesmo efeito [79]:

```
TempoEmMilissegundos := 500
Sleep, TempoEmMilissegundos
Sleep, %TempoEmMilissegundos%
Sleep, % TempoEmMilissegundos
```

Apesar de possuir alguns aspectos negativos em relação a sua sintaxe, podemos afirmar que o grande diferencial do AHK está nas suas três funcionalidades mais notórias: teclas de atalho (globais e sensíveis a contexto), remapeamento de teclas intuitivo e expansão/substituição de texto através de “*hotstrings*”.

A simplicidade com a qual é possível atribuir ações a determinadas combinações de teclas é provavelmente a característica mais conhecida da linguagem. Com opções sensíveis ao contexto e uma vasta flexibilidade, quase qualquer tecla de atalho pode ser mapeada [84,85]. Buscamos exemplificar de forma sucinta no script abaixo uma série de atribuições, a fim de ressaltar a facilidade de criá-las.

```
a:: MsgBox, Apertou a tecla 'a' sozinha
^f:: MsgBox, Apertou CTRL+f
*t:: MsgBox, Apertou a tecla 't' (sozinha ou não!)
+Lbutton:: MsgBox, Apertou SHIFT + botão esquerdo do mouse
b & c:: MsgBox, Apertou as teclas 'b' e 'c' juntas
#IfWinActive, ahk_class Notepad ; sensível ao contexto
!s:: MsgBox, Apertou ALT+s em uma janela do Bloco de Notas
```

Além de apresentarem uma sintaxe extremamente simples para a funcionalidade que proporcionam, os atalhos interpretados pelo AutoHotkey podem disparar fluxos de execução mais complicados, como mostramos no exemplo abaixo:

```
#n:: ; combinação de teclas: botão do Windows + n
    Run Notepad ; abre uma nova instância do Bloco de Notas
    WinWaitActive, ahk_class Notepad
    SendRaw, % "Olá usuário!" ; digita na janela ativa
    Sleep, 1000
    WinMaximize, A ; maximiza a janela ativa
    MsgBox, Maximizado!
Return ; finaliza o fluxo de execução disparado pelo atalho
```

O remapeamento de teclas é realizado de maneira similar, utilizando apenas uma linha de código e funcionando até mesmo com combinações de teclas [86] O remapeamento pode ser implementado até mesmo para desabilitar uma tecla, como mostra a última linha do próximo exemplo.

```
a::b ; a tecla 'a' funciona como a tecla 'b'
e::f
f::e ; combinada à linha acima, troca as teclas 'e' e 'f'
MButton::Shift ; botão do meio do mouse funciona como SHIFT
RButton::+LButton ; Mouse_Direito ➔ SHIFT + Mouse_Esquerdo
```

```
^x::^c ; CTRL+x ➔ CTRL+C (CTRL+ALT+x ➔ CTRL+ALT+c, etc)
#IfWinActive, ahk_class Notepad ; sensível ao contexto
p::Return ; desabilita a tecla 'p' (no Bloco de Notas)
```

Outra funcionalidade com muito potencial em automações rotineiras são as chamadas *hotstrings* – sequências de caracteres que podem ser detectadas pelo AHK quando digitadas, permitindo disparar uma substituição/expansão de texto ou mesmo um fluxo de execução [87]. Exemplificamos abaixo algumas destas opções de uso:

```
; Substituir o texto "opcao" por "opção" quando digitado e
; seguido de um finalizador (pontuação, espaço, Enter...)
::opcao::opção
; Posicionar o cursor dentro de tag HTML "em": <em>|</em>
:*b0:<em>::</em>{left 5}
; Adicionar o texto "def" quando o texto "abc" for digitado
; mesmo estando no meio de uma palavra e sem a necessidade
; de um finalizador
:?*:abc::def
; Substituir (utilizando as mesmas opções acima) o texto
"123" pelo texto "456" seguido de um Enter e do texto "789"
:?*:123::
(
456
789
)
; Abrir uma janela do Bloco de Notas ao digitar "/bloco"
:?*:/bloco::
    Run, Notepad
Return
```

Como já citamos no início da seção 2.1, o AHK é capaz de realizar inúmeras tarefas. Exibimos acima alguns trechos de código apenas para fins ilustrativos (especialmente da sintaxe). Não temos a intenção de lecionar aqui a linguagem por completa – para isso, recomendamos que a documentação da própria ferramenta seja consultada, pois ela apresentará sempre um referencial mais atualizado.

2.2 Comparação com Softwares Similares

Não seria correto elaborar um trabalho sobre o AutoHotkey sem mencionar a existência de alguns softwares relacionados e fazer algumas comparações que busquem inclusive justificar o uso do AutoHotkey em primeiro lugar. Optamos por nos referir a tais softwares como "alternativas" ao invés de "concorrentes", pois o uso de um deles não necessariamente impede o uso paralelo de outro. Entretanto, como alguns softwares possuem múltiplas funcionalidades em comum, é razoável compreender que a utilização de um deles em certos cenários pode dispensar a necessidade de outros.

2.2.1 Alternativas

Embora seja possível encontrar na internet uma quantidade considerável de alternativas, optamos por deixar de fora de nossas comparações softwares como o "JitBit Macro Recorder" [87] que não sejam 100% gratuitos e softwares como o "Action(s)" [88] que não estejam em desenvolvimento ativo ou que não possuam código aberto. A única exceção dentro destes critérios é o AutoIt, inclusive por questões históricas relacionadas ao surgimento do AutoHotkey (como visto na seção 2.1.1.1).

Vale observar também que não incluímos em nossa lista de alternativas o Selenium – um software idealizado em 2004 como framework de testes para aplicações web [89,90]. Mesmo sendo uma ferramenta conhecida e robusta para gravação de macros e criação de testes [91], seu escopo é limitado, servindo apenas para gravar e executar ações internas a navegadores (como Internet Explorer, Firefox e Chrome). Por este motivo, argumentamos que não há sentido em compará-lo com outros programas de automação que são capazes de interagir com todo o sistema operacional.

2.2.1.1 AutoIt

AutoIt é um software para automação de tarefas em Windows implementado também através de uma linguagem interpretada e desenvolvido desde o final de 1998. Seu criador – Jonathan Bennett – utilizava uma antiga ferramenta do Windows (indisponível há muitos anos) chamada "ScriptIt" [92-94], mas não considerava a ferramenta muito robusta e sentia falta de outras funcionalidades. Assim, o AutoIt surgiu inicialmente como uma alternativa mais robusta e completa em relação ao ScriptIt para realizar tarefas padronizadas (como instalações e configurações) em qualquer computador com Windows [95].

Até o final de 2002, a linha de desenvolvimento do programa atingiu a versão 2 (“v2”), o código foi reescrito em C++ e diversas funções foram adicionadas. Em 2003, Bennett começou a perder o interesse no projeto pois, entre outros motivos, considerava muito complicada a manutenção de seu código. Pressionado por outros usuários, o autor optou por reformular boa parte do AutoIt, transformando-o em uma linguagem de script mais completa e culminando no surgimento oficial do AutoIt v3 no início de 2004 [96,97]. Depois de expressar em 2005 seu descontentamento com derivações do projeto (que utilizava até então uma licença de código aberto), Bennett mudou a licença do software a partir da versão 3.2.0 (em agosto de 2006) para uma licença fechada [98,99].

Por ser o software que serviu como inspiração para o AHK, o AutoIt apresenta diversas características e funcionalidades semelhantes, como desempenho, portabilidade, baixo consumo de memória, compilação de scripts, suporte a sistemas de 64 bits e criação de interfaces gráficas. Em seu instalador, é incluída uma versão mais simplificada da IDE sugerida no website oficial para criação e edição de scripts (SciTE4AutoIt3), como ilustra a Figura 1 [95,100,101].

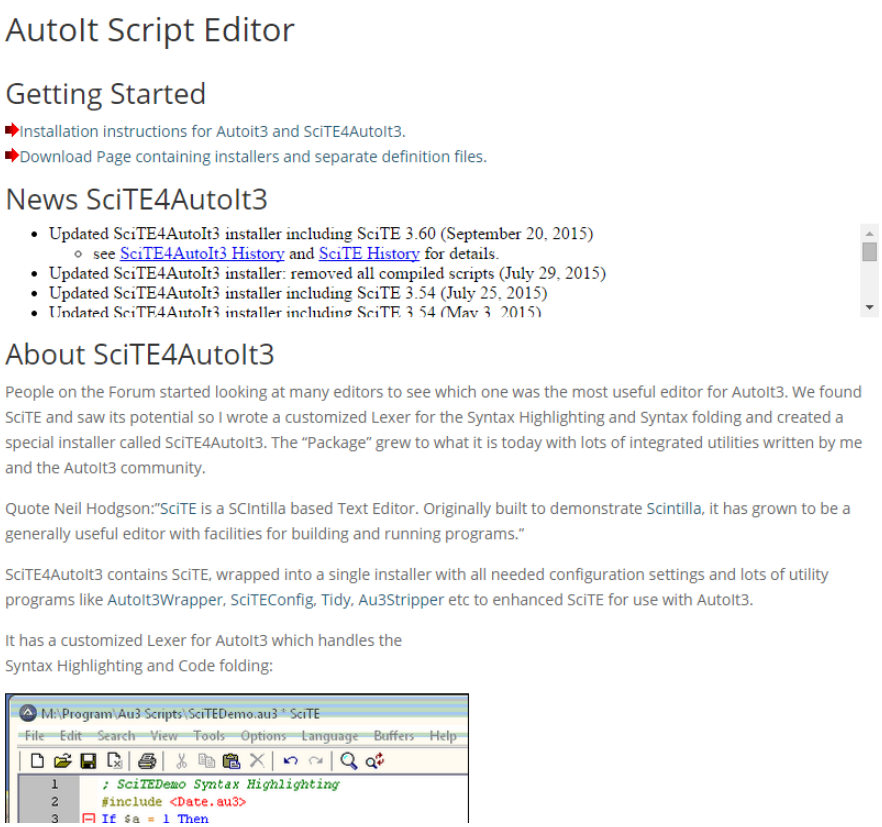


Figura 1: Página sobre o “AutoIt Script Editor” em *autoitscript.com*

2.2.1.2 SikuliX

Sikuli é um software de automação criado em Java cujo desenvolvimento foi iniciado em 2009 por Tsung-Hsiang Chang e Tom Yeh – membros à época do *User Interface Design Group* no Instituto de Tecnologia de Massachusetts (MIT) [102]. A última versão do projeto original foi lançada em 2011, antes de ser abandonado por seus criadores no ano seguinte. Interessado no projeto, Raimund Hocke continuou seu desenvolvimento a partir de 2013, utilizando o nome SikuliX [103-106].

Sua principal e mais conhecida característica – que o distingue de outros softwares de automação – é o uso de reconhecimento visual para identificar elementos na tela do computador a fim de guiar o fluxo de execução de scripts. Embora tal funcionalidade esteja presente tanto no AutoHotkey como no AutoIt (através do comando/função "ImageSearch"), o verdadeiro diferencial está no Sikuli IDE – um editor de scripts nativo, que facilita a captura de tela e permite a manipulação imediata de tais capturas, que aparecem no código como imagens. Assim, não há necessidade de salvar manualmente as imagens e torna-se muito mais fácil compreender qual imagem está sendo utilizada em cada trecho do código, como ilustrado na Figura 2 [107].

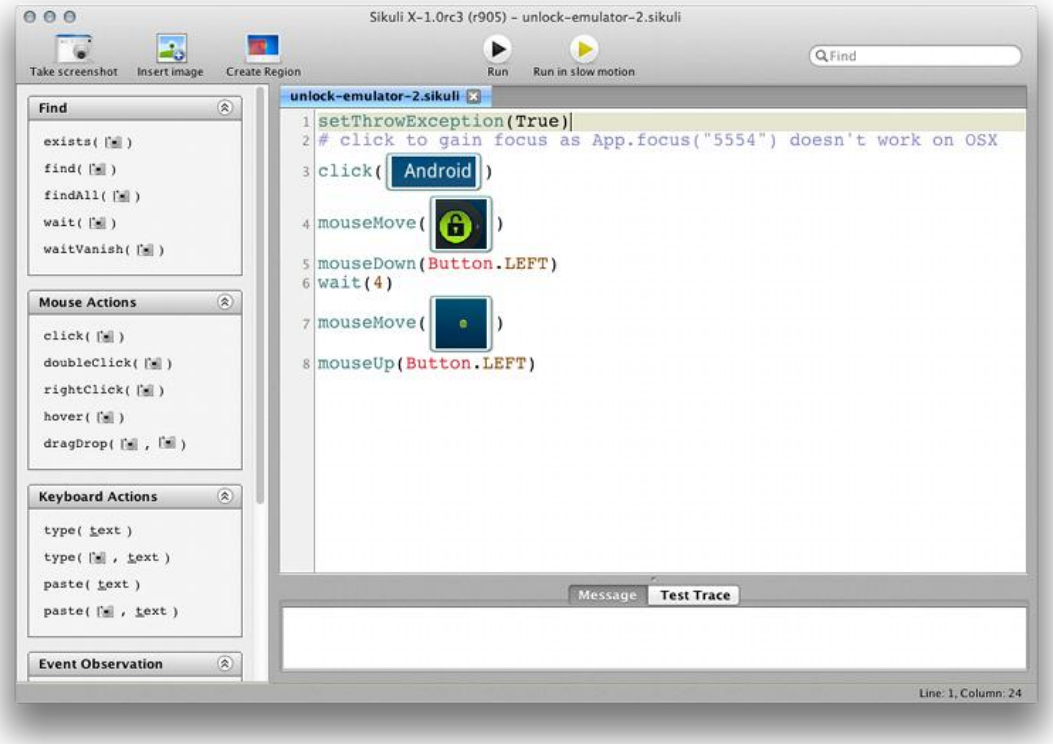
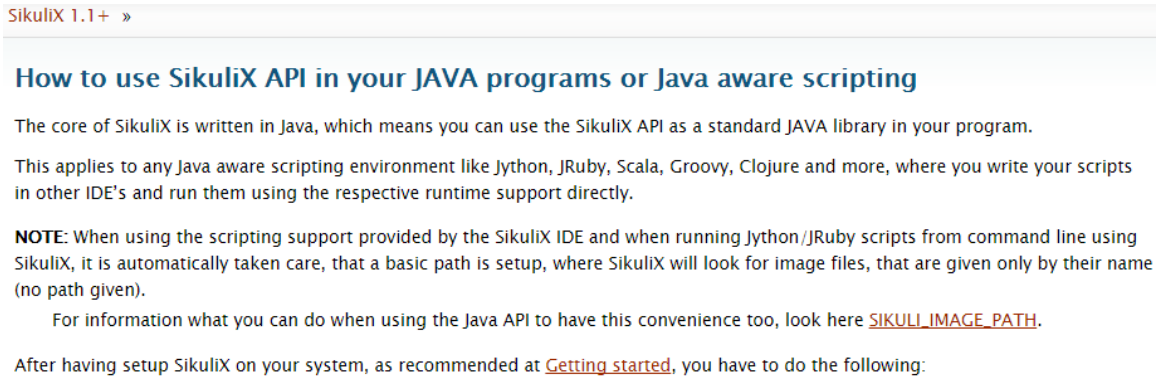


Figura 2: Exemplo de visualização de script no "Sikuli IDE"

Apesar de fornecer algumas funções adicionais específicas, o SikuliX permite o uso das linguagens Python e Ruby para a criação de scripts. Informações de uma página mantida por Hocke (ilustrada na Figura 3) sugerem que outras linguagens (como Java, Jython, JRuby, Scala e Clojure) podem ser utilizadas. Entretanto, com outras fontes desatualizadas na internet, a documentação da ferramenta é esparsa e confusa, dificultando o encontro dos detalhes e exemplos mais corretos [108-110].


SikuliX 1.1+ »

How to use SikuliX API in your JAVA programs or Java aware scripting

The core of SikuliX is written in Java, which means you can use the SikuliX API as a standard JAVA library in your program.

This applies to any Java aware scripting environment like Jython, JRuby, Scala, Groovy, Clojure and more, where you write your scripts in other IDE's and run them using the respective runtime support directly.

NOTE: When using the scripting support provided by the SikuliX IDE and when running Jython/JRuby scripts from command line using SikuliX, it is automatically taken care, that a basic path is setup, where SikuliX will look for image files, that are given only by their name (no path given).

For information what you can do when using the Java API to have this convenience too, look here SIKULI_IMAGE_PATH.

After having setup SikuliX on your system, as recommended at Getting started, you have to do the following:

Figura 3: Página em *readthedocs.org* sobre compatibilidade com outras linguagens

2.2.2 Breve Análise Comparativa

Em busca de outras fontes mais formais que tenham feito análises deste tipo, encontramos uma literatura que realiza uma comparação entre AHK, AutoIt e Sikuli [111]. Entretanto, as informações encontram-se desatualizadas, pois este tipo de levantamento pode ter um prazo de validade muito curto. Isso se justifica não apenas pela baixa frequência (ou mesmo ausência) de trabalhos acadêmicos específicos sobre tais softwares, mas também pela rápida evolução no desenvolvimento destes programas.

Assim, é importante lembrar que este trabalho eventualmente também não representará o estado atual de cada ferramenta e as referências online devem ser acessadas para uma caracterização mais fiel dos softwares. Existem hoje inclusive propostas para uma versão 2 tanto do AutoHotkey [112] como do SikuliX [113,144], que mudarão drasticamente diversos conceitos quando lançadas oficialmente.

Desde a criação do AutoHotkey – muito inspirada no AutoIt – existe uma certa "rivalidade" entre estas duas ferramentas [115], sendo comum observar na internet diversas comparações informais entre os softwares [116-119]. Por apresentarem várias semelhanças, muitos dos critérios usados para tais avaliações são subjetivos, como a facilidade de aprender a linguagem ou a hospitalidade da comunidade online com usuários iniciantes. Sendo assim, buscamos ser objetivos na construção da tabela 1,

selecionando atributos que ajudem a diferenciar as características e funcionalidades de cada software.

	SikuliX	**AutoIt**	**AutoHotkey**
Licença de Software atual	MIT	Licença fechada	GNU GPL v2
Sistemas Operacionais suportados	Windows, Linux, OS X	Windows	Windows
Orientação a Objetos nativa	Não	Somente *arrays*	Sim
Ambiente de Desenvolvimento Integrado (IDE) sugerido	Sikuli IDE	SciTE4AutoIt3	SciTE4AutoHotkey
Pré-requisitos a serem instalados para executar algum script da ferramenta	Java, SikuliX[108]	Nenhum	Nenhum
Teclas de Atalho Globais	Sim[120]	Sim[121]	Sim[84,122]
Teclas de Atalho Sensíveis a Contexto	Não	Não[123]	Sim[84,122]
Remapeamento de Teclas	Não	Não	Sim
Expansão de Texto	Não	Não	Sim

Tabela 1: Comparação entre Sikuli, AutoIt e AutoHotkey

Através desta análise por critérios objetivos, concluímos que o AutoHotkey possui mais funcionalidades de uma maneira geral, sem apresentar nenhum tipo de desvantagem severa em relação a outros softwares. Justificamos, portanto, nossa escolha de ferramenta para a realização deste trabalho.

3 Estudo De Caso

3.1 Objetivo

Para aplicar na prática a utilização do AutoHotkey a fim de demonstrar seu potencial como ferramenta de automação, optou-se por buscar usuários próximos do ambiente acadêmico que utilizassem um computador rotineiramente para suas atividades. Em conversa com o principal funcionário da secretaria do Bacharelado em Sistemas de Informação (BSI) da UNIRIO responsável por atender os alunos, o mesmo revelou que boa parte de suas tarefas diariamente executadas em seu computador de trabalho envolviam o uso do Sistema de Informações para o Ensino (SIE). O SIE é um software para gestão integrada que propõe a integração de todas as atividades de uma instituição de ensino superior, permitindo a gestão de informação através da integração de módulos. A UNIRIO tem implantados os módulos "Acadêmico" (Graduação e Pós-Graduação) e "Recursos Humanos" (Cadastro Básico) [124].

3.2 Escolha da Atividade

O funcionário, que trabalha na secretaria há pelo menos 7 anos, citou especificamente que gasta muito tempo a cada início de período letivo para matricular os alunos ingressantes do BSI em suas devidas disciplinas – tipicamente, cadastrar cada um dos 36 alunos em 6 turmas. Os próprios alunos devem efetuar suas inscrições em disciplinas utilizando outro sistema a partir do segundo período, mas para as disciplinas do primeiro período esse processo precisa ser efetuado pelo funcionário. Embora nunca tenha cronometrado essa atividade, sua estimativa era de pelo menos 45 minutos para completá-la, considerando que este tempo poderia aumentar bastante devido à necessidade de interromper o processo para atender alunos pessoalmente, organizar papeladas, entre outras atividades que demandavam sua atenção imediata.

Com base nesse levantamento, ficou claro que tal atividade seria o alvo de nosso estudo de caso, pois sua automação traria a maior recompensa para o funcionário em termos de tempo economizado e redução de esforço.

3.3 Entendimento do Problema

O início do trabalho prático consistiu na ambientação com o sistema utilizado, analisando seu funcionamento durante o processo de inscrição dos alunos de primeiro período. A Figura 4 mostra a principal tela do SIE disponível para o funcionário, que permite a navegação entre diversas funcionalidades. A funcionalidade em destaque – "Matrícula por Aluno" – é utilizada pelo funcionário para efetuar alterações nas inscrições de alunos em disciplinas. Ao clicar duas vezes nesta opção selecionada, a tela representada pela Figura 5 é exibida. É a partir da presença desta tela que temos interesse de automatizar nosso processo.

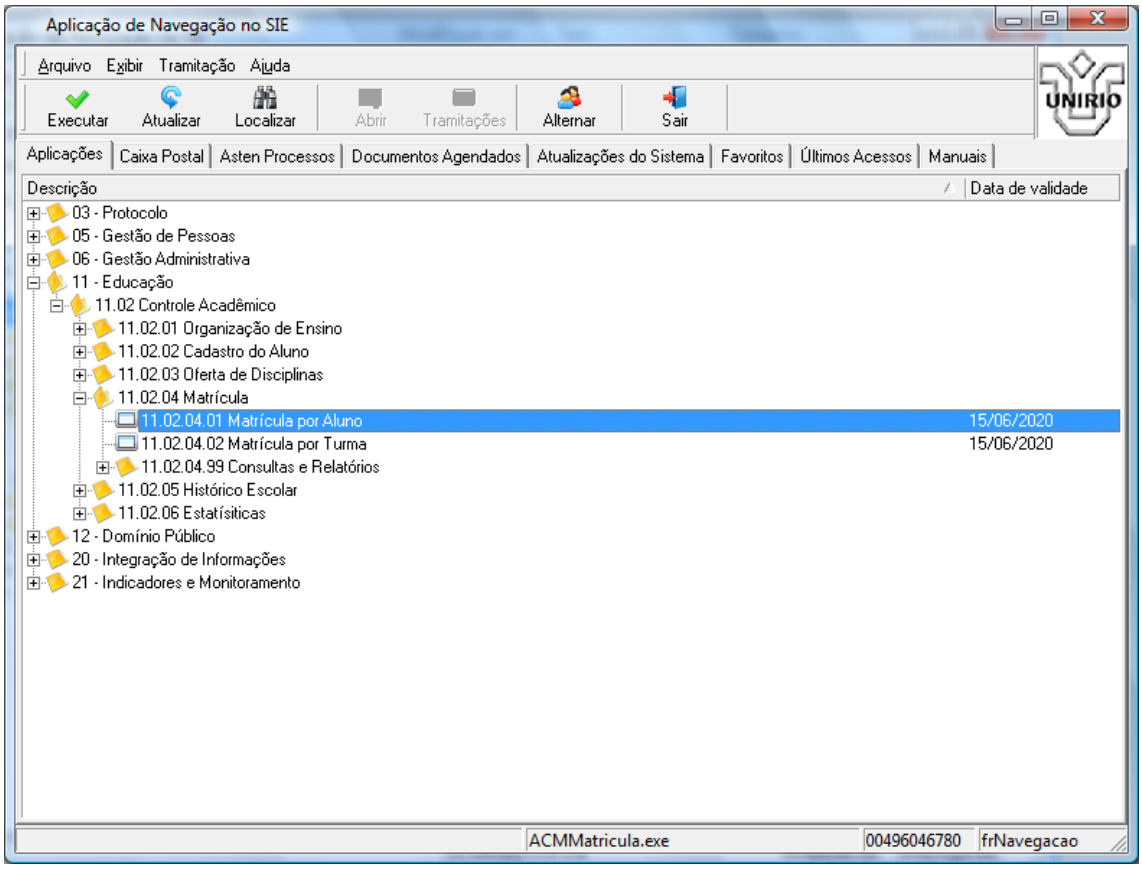


Figura 4: Janela principal da "Aplicação de Navegação do SIE"

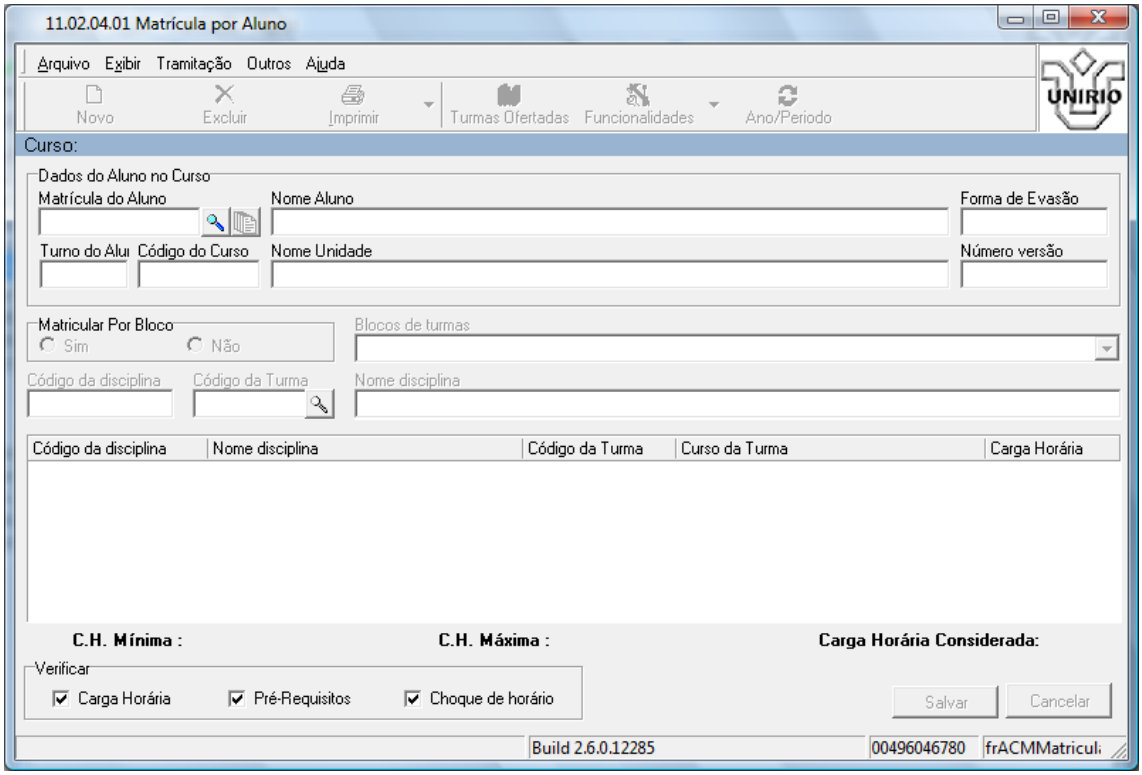


Figura 5: Janela principal da funcionalidade "Matrícula por Aluno"

Ao clicar no botão (contendo a imagem de uma lupa) próximo ao texto "Matrícula do Aluno", a tela da Figura 6 é apresentada, onde o funcionário costuma inserir o número completo da matrícula desejada e clicar no botão "OK". Apertar a tecla "Enter" nesta tela tem o mesmo efeito que apertar o botão. Com isso, a janela desaparece e a tela anterior é atualizada, conforme visto na Figura 7.

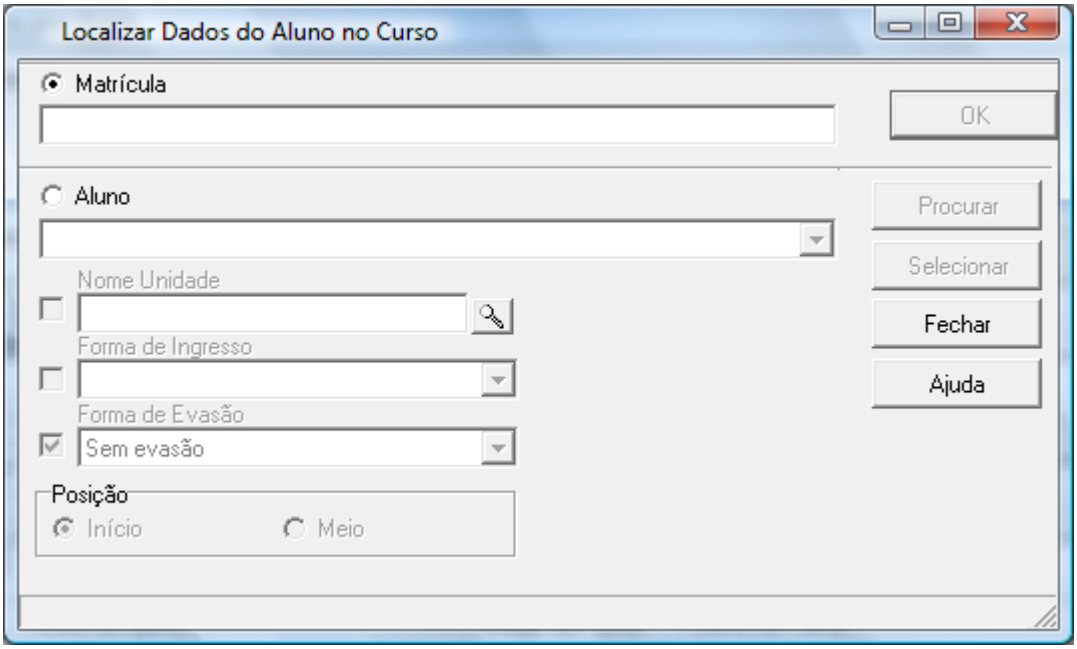


Figura 6: Janela para localização e seleção de aluno / matrícula

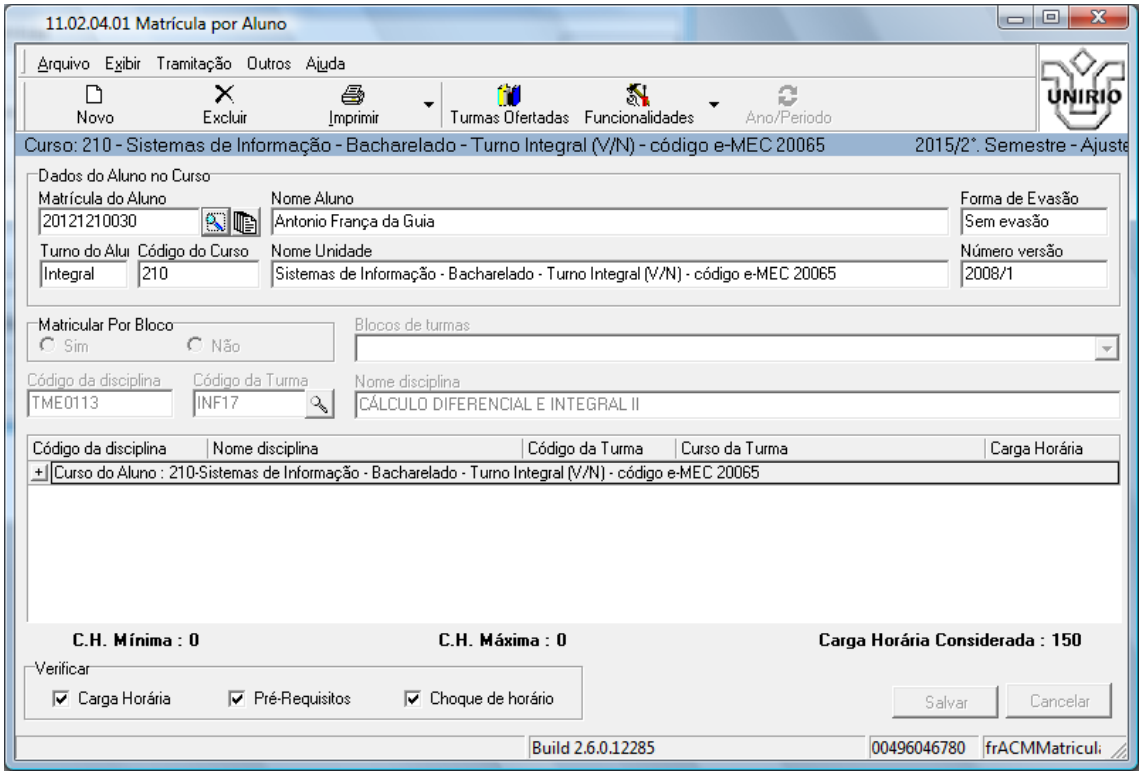


Figura 7: Janela "Matrícula por Aluno" atualizada com a matrícula selecionada

A partir deste estado, as próximas ações terão efeito sobre a última matrícula selecionada. Essa observação é importante, pois significa que as etapas anteriores não precisam ser repetidas até que o aluno esteja inscrito nas 6 turmas desejadas. Para iniciar a adição de uma disciplina, é necessário clicar no grande botão "Novo" próximo ao canto superior esquerdo da tela. Assim, o segundo botão com a imagem de uma lupa (próximo ao texto "Código da Turma") é habilitado, confirme exibido pela Figura 8.

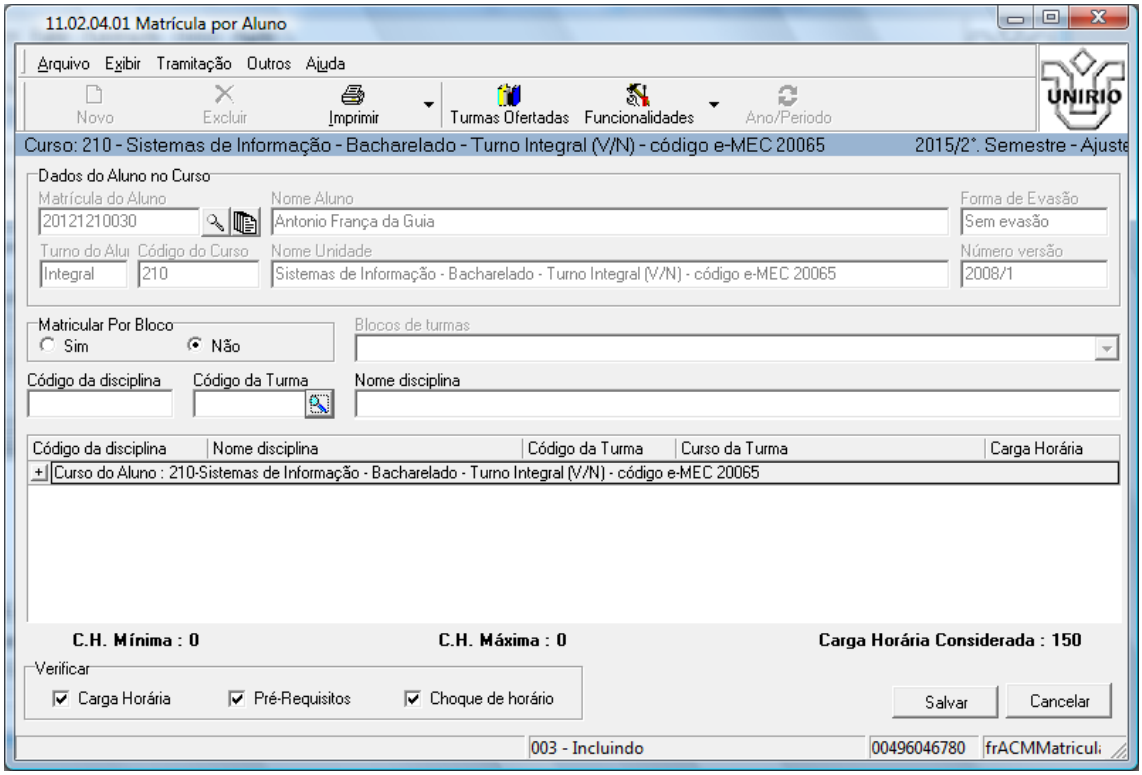


Figura 8: Janela "Matrícula por Aluno" em estado de nova inscrição

Clicando no segundo botão de lupa mencionado acima, aparece para o usuário uma tela chamada "Localizar Turma". Esta tela aparece "vazia" (como na Figura 9) se nenhuma busca por turma foi realizada desde a seleção da funcionalidade da Figura 4. Caso contrário, a mesma tela pode aparecer com algumas informações já preenchidas, como exemplificado na Figura 10. Esse detalhe também é importante, pois alterações na forma como a tela é apresentada podem impactar diretamente no script de automação, exigindo uma lógica de detecção e interação com a tela mais detalhada ou até completamente diferente.

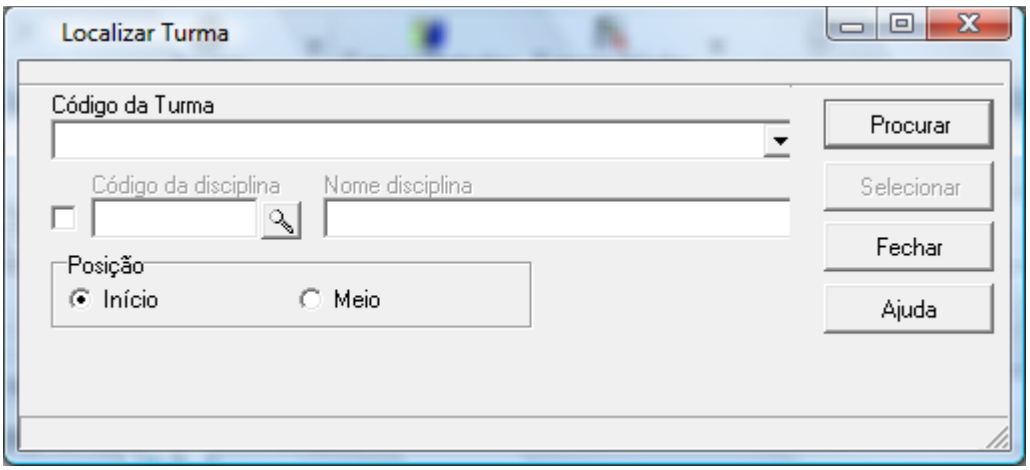


Figura 9: Janela para localização e seleção de turmas "vazia"

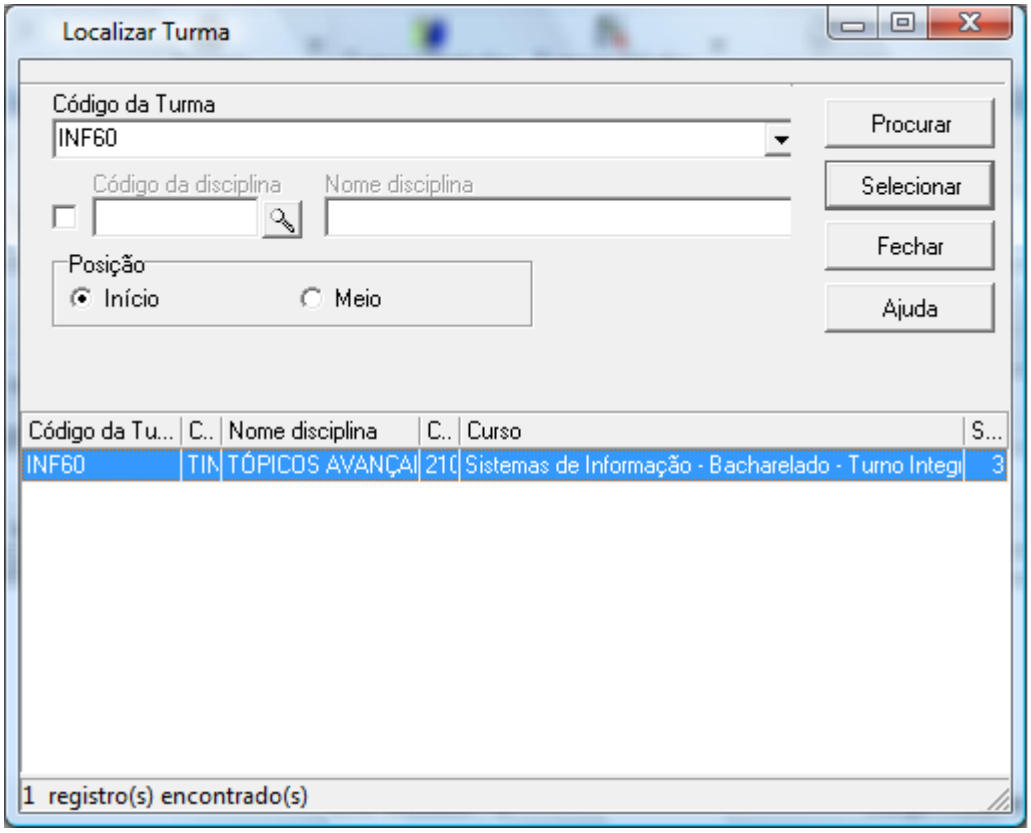


Figura 10: Janela para localização e seleção de turmas "preenchida"

Independente do estado da tela anterior, o usuário deve inserir um código de turma e clicar no botão "Procurar" (ou apertar a tecla "Enter"). A listagem de turmas na tela é então atualizada, omitindo turmas referentes a disciplinas já cursadas pelo aluno. O usuário pode então selecionar uma ou mais turmas e clicar no botão "Selecionar". Clicar duas vezes em uma única turma da listagem produz o mesmo efeito, selecionando apenas aquela turma e dispensando a necessidade de clicar no botão.

Dependendo do termo buscado, a listagem nesta tela pode exibir qualquer número de resultados, inclusive zero. Vale notar que existe uma "barra de status" no "rodapé" da tela exibindo a quantidade de resultados encontrados – informação que pode ser útil para a automação desta etapa. Quando efetuado um clique no botão "Selecionar" ou um clique duplo em uma turma, a janela atual é fechada e a tela anterior (Figura 8) é atualizada, adicionando a(s) devida(s) turma(s) na lista de disciplinas.

Ao clicar no botão "Salvar", o sistema verifica possíveis conflitos de horário entre disciplinas e outras condições importantes – exibidas no canto inferior esquerdo da tela e que, de acordo com o funcionário da secretaria, ficam sempre selecionadas por padrão. Se nenhuma verificação encontrar problemas, o sistema processa o armazenamento das

informações por algum tempo (tipicamente menos de 2 segundos) e retorna a tela para o estado exibido pela Figura 11. Este estado é basicamente o mesmo da Figura 7, onde o usuário pode clicar no botão “Novo” para iniciar a adição de uma disciplina ou clicar no primeiro botão de lupa para selecionar um aluno. Vale ressaltar que a única diferença caso o botão “Cancelar” seja clicado ao invés do botão “Salvar” é que os passos de verificação e armazenamento não são executados.

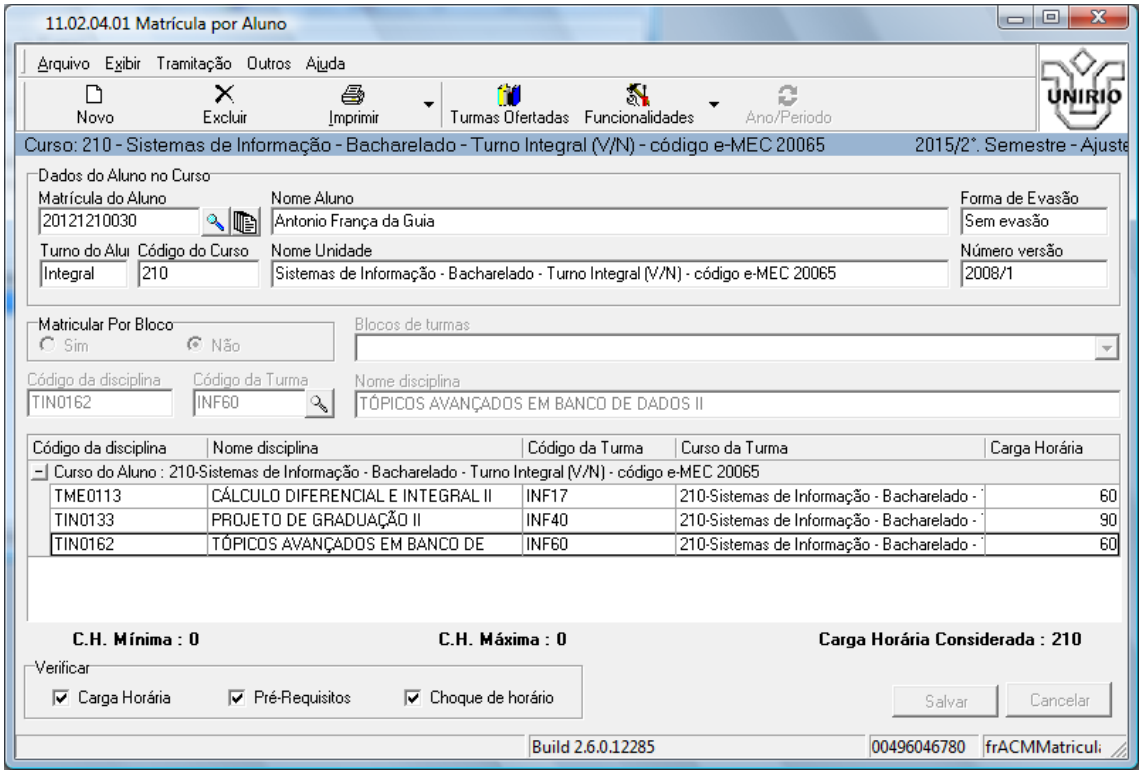


Figura 11: Janela de “Matrícula por Aluno” atualizada com turma(s)

3.4 Desenvolvimento da Solução

Conhecendo todo o procedimento descrito anteriormente, foi possível resumi-lo de maneira genérica nas seguintes grandes etapas:

1. Selecionar uma matrícula
2. Selecionar uma turma
3. Salvar a inscrição da matrícula selecionada na turma selecionada
4. Pular para a etapa 2 até que o aluno esteja inscrito em todas as disciplinas
5. Pular para a etapa 1 até que todos os alunos estejam inscritos devidamente

Com tal estrutura organizada e separada em etapas distintas, ficou clara a existência de pelo menos duas grandes repetições no procedimento. Ao longo de nosso trabalho, essa compreensão permitiu a criação de módulos em nosso script, facilitando inclusive a refatoração e manutenção do código.

Antes de começar a codificar nossa automação, optou-se por instalar o AutoHotkey no computador da secretaria para agilizar nossos testes. Em seguida, seria necessário justamente testar a interação do AHK com o SIE, a fim de descobrir se as janelas da aplicação responderiam corretamente à execução de funções do AHK como "WinExist" e comandos como "WinActivate" e "MouseClick". As tentativas iniciais – utilizando scripts bastante simples que tentavam ativar a janela "Matrícula por Aluno" e clicar em regiões da tela – acabaram falhando. A ativação de janelas não funcionava todas as vezes e os cliques de mouse gerados artificialmente pelo AHK pareciam não funcionar nos "controles" (elementos visuais de uma aplicação típica de Windows, como botões, campos para digitação de texto, listas e caixas de seleção).

Utilizando o "Active Window Info" (software "espião de janela" mencionado anteriormente e ilustrado na Figura 12, que acompanha a instalação do AutoHotkey), foi possível verificar que os controles e as próprias telas apresentadas pelo SIE possuíam uma classe interna no Windows (identificada para o AHK pelo atributo "ahk_class" [125]) iniciada com a letra maiúscula "T" – como "TBitBtn2", "TPanel1" e "TfrCpErrorsVisual". Pesquisando brevemente por dicas sobre este tipo específico de nomenclatura, ficou claro que a aplicação (ou ao menos sua interface gráfica) foi desenvolvida em Delphi [126-128]. Assim, levantou-se a suspeita de que o AutoHotkey não seria capaz de interagir de maneira robusta e confiável com aplicações deste tipo.

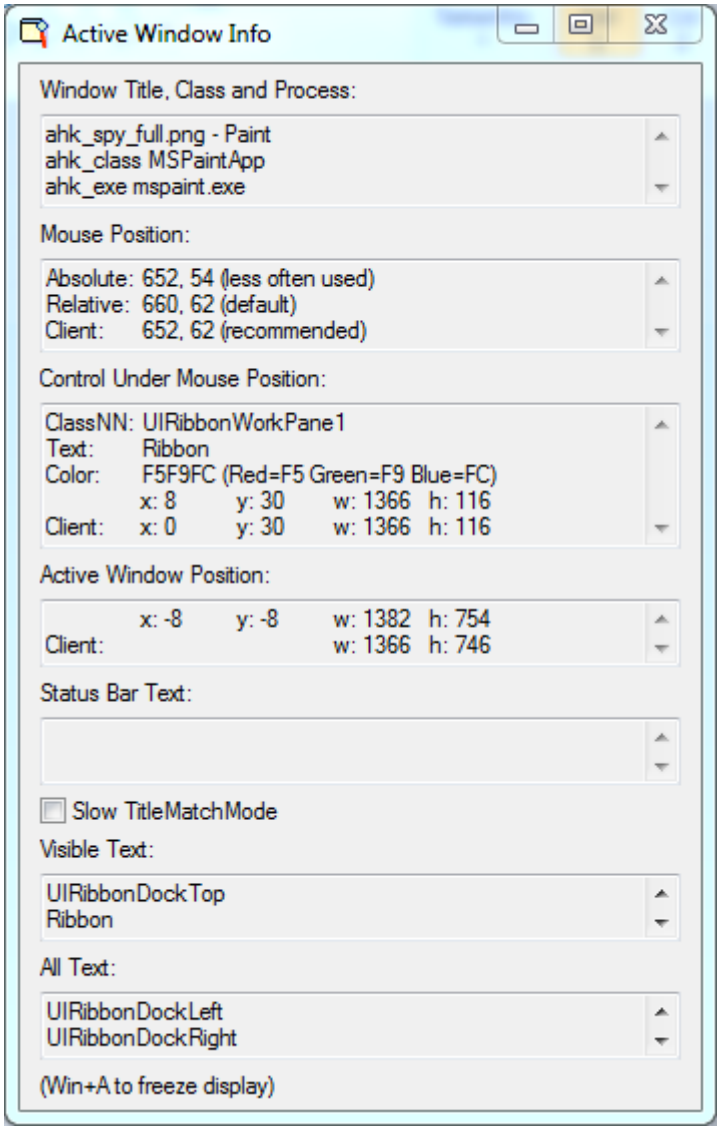


Figura 12: "Active Window Info" – software "espião de janela"

Após mais algumas buscas sobre o assunto nos fóruns oficiais do AutoHotkey, constatamos que outras pessoas relatavam automações bem sucedidas envolvendo aplicações criadas com Delphi. Tentamos ainda revisar nossos próprios scripts de teste, na esperança de identificar algum erro ou a ausência de alguma instrução importante. Por fim, recordamos que algumas pessoas na internet eventualmente mencionam a necessidade de executar o interpretador do AHK como administrador da máquina – uma funcionalidade do Windows que concede permissões mais amplas para um aplicativo durante sua execução. Ao habilitar esta opção, todos os nossos testes passaram a apresentar o funcionamento esperado.

Com as telas do SIE respondendo às ações do AutoHotkey, iniciamos a criação de nosso script codificando diversas funções auxiliares importantes, que seriam úteis em

diversos pontos da execução. Isso nos trouxe diversas vantagens relacionadas ao uso de programação modular, evitando o reuso de código e facilitando a codificação e compreensão do script como um todo. Listamos a seguir algumas de nossas funções auxiliares mais importantes:

- "AtivarTela" – garante que uma janela especificada por parâmetro seja ativada e retorna identificador único da janela no Windows, arremessando uma exceção se o tempo limite for excedido ou se algum comando falhar

- "EsperarTelaFechar" – não fecha nenhuma janela, mas espera (também com um limite de tempo) que tal janela não exista mais no ambiente do usuário (nem mesmo minimizada ou escondida atrás de outras janelas)

- "Clicar" – invoca alguns comandos adicionais além de "MouseClick" para garantir que movimentações acidentais do mouse durante a execução desse comando não resultem em um clique fora do local desejado

- "VerificarTelaDeErro" – quando chamada, verifica no ambiente do usuário a presença de um tipo específico de tela (referente a erros no SIE) e arremessa uma exceção caso a presença seja confirmada

- "MsgBox" – invoca o comando de mesmo nome, mas garante antes que um comando "BlockInput, On" possivelmente executado pela função "Clicar" não impeça o usuário de interagir com a caixa de diálogo exibida pelo AutoHotkey

- "Abortar" – prepara um texto detalhado a ser exibido antes de finalizar o programa, com possíveis instruções para o usuário sobre como ele deve prosseguir dependendo dos valores dos parâmetros utilizados (ilustrado na Figura 13)

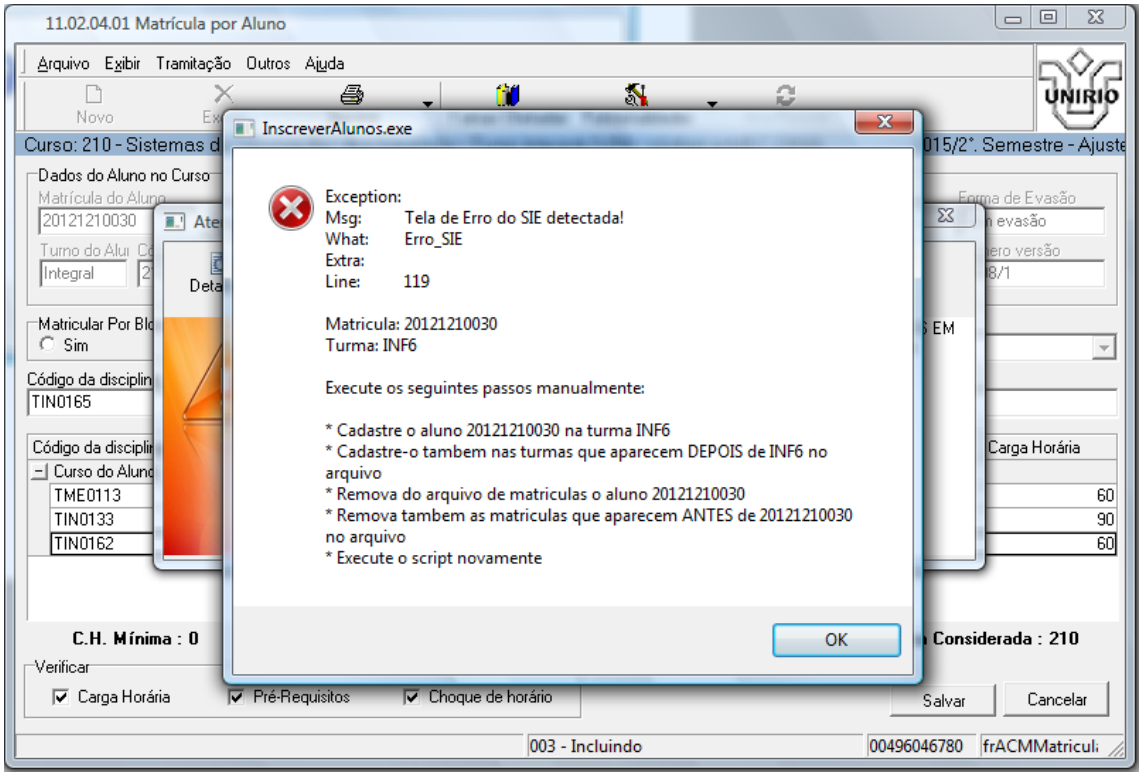


Figura 13: Mensagem de erro exibida pelo script

Ao longo da criação de tais funções e dos testes realizados, anotamos também as coordenadas relevantes do cliques artificiais que precisariam ser simulados pelo script. Todas as coordenadas foram descobertas com o "Active Window Info" utilizando a opção "Client", que não apenas é referente à janela ativa, como também é menos dependente da versão do sistema operacional e de temas gráficos adotados pelo usuário [129]. Assim, execuções futuras do script não são afetadas se o usuário desejar mover, diminuir ou expandir alguma janela do SIE. Além disso, aproveitamos para nos assegurar de que todos os testes seriam finalizados clicando no botão "Cancelar" até que nossa aplicação estivesse quase finalizada e confiável o suficiente para a realização de testes com o botão "Salvar".

Vale ressaltar também que configuramos a tecla "ESC" como um atalho rápido para abortar o script a qualquer momento. Esta é uma dica fundamental para praticamente qualquer teste ou mesmo script funcional, pois o controle do AutoHotkey sobre o mouse e o teclado pode dificultar ou até mesmo impossibilitar o encerramento do programa sem a necessidade de uma ação mais drástica (como o desligamento do computador). Este atalho foi inclusive mantido na versão final de nossa aplicação, pois argumentamos que fornecer ao usuário uma possibilidade de finalizar imediatamente o script pode ser considerado uma boa prática.

Neste ponto do trabalho, o script executava dois comandos “Loop” – um dentro do outro. O mais externo selecionava 36 vezes um mesmo número fixo de matrícula e o mais interno trabalhava 6 vezes com um código de turma também fixo. Como precaução, decidimos inserir diversas pausas ao longo do script para que sua execução não fosse muito rápida, pois poderíamos correr o risco de disparar uma interação sem que a tela do SIE terminasse de processar a interação anterior. Nosso código estava repleto de linhas executando o comando “Sleep” com valores relativamente altos e alguns outros comandos – como “WinWait” – também estavam configurados com limites de tempo elevados, como 10 segundos ou mais.

Por conta dos valores fixos citados acima, cogitamos algumas possibilidades para flexibilizar o uso de matrículas e turmas diferentes em cada iteração. Consultando o funcionário da secretaria, verificamos que seria fácil obter (ou mesmo criar) sem grande esforço um arquivo de texto com um número de matrícula por linha. Devido à pequena quantidade de turmas, a mesma estratégia poderia ser usada para armazenar a lista de códigos de turma. Desenvolvemos então uma solução que carrega o conteúdo de dois arquivos localizados em disco no mesmo local do script, chamados “matriculas.txt” e “turmas.txt”. Estes conteúdos são limpos para evitar problemas de formatação (eliminando linhas em branco, espaços e outros caracteres desnecessários) e processados linha por linha em seus respectivos loops – sendo o loop externo referente a matrículas e o loop interno referente a turmas, como visto anteriormente.

Para aprimorar o tempo de execução do script, passamos a eliminar diversas chamadas do comando “Sleep”. Nas chamadas restantes, reduzimos o tempo de espera utilizado como parâmetro. Após a realização de mais teste, atingimos um balanço satisfatório entre a velocidade de execução e a confiabilidade do programa. Adicionando algumas linhas de código para cronometrar todo o processamento realizado, constatamos que a simulação – inscrição de 36 alunos em 6 disciplinas cada – era finalizada com sucesso em cerca de 18 minutos. A caixa de diálogo preparada para essa medição de tempo (ilustrada na Figura 14) foi inclusive mantida no produto final, sendo exibida no término da execução. Como última etapa do desenvolvimento do script, realizamos testes finais utilizando o botão “Salvar” na aplicação do SIE. Para evitar conflitos com as checagens do sistema, também implementamos rapidamente depois de cada clique neste botão uma rotina responsável por excluir a última inscrição do aluno. A conclusão dos testes indicou um tempo decorrido de 24 minutos.

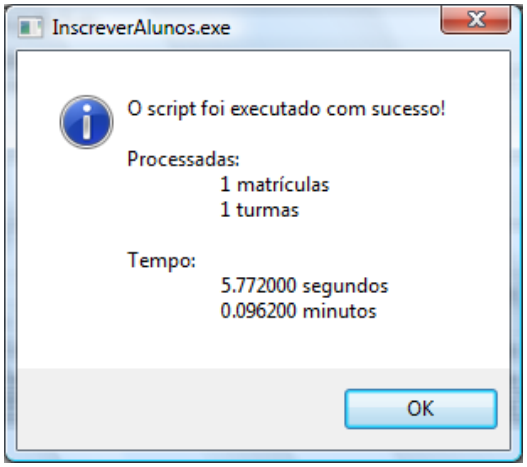


Figura 14: Mensagem de sucesso exibida pelo script

Finalmente, removemos do código a rotina de exclusão e utilizamos uma pequena quantidade de dados (cerca de 3 matrículas e 2 turmas) para garantir que o script funcionaria corretamente. Com base nos resultados anteriores, estimamos que processamentos futuros com a quantidade total de dados sejam realizados dentro de aproximadamente 20 minutos. Consideramos assim que encerramos com sucesso a automação da atividade e pretendemos dar suporte ao uso de nosso programa no próximo início de período letivo (previsto para fevereiro ou março de 2016).

A Figura 15 representa o modelo final da solução, com três arquivos que devem ficar no mesmo local em disco. Além do arquivo executável com a versão compilada do script, são necessários também os dois arquivos de texto – mencionados anteriormente – para o armazenamento das matrículas e das turmas a serem processadas pelo script.

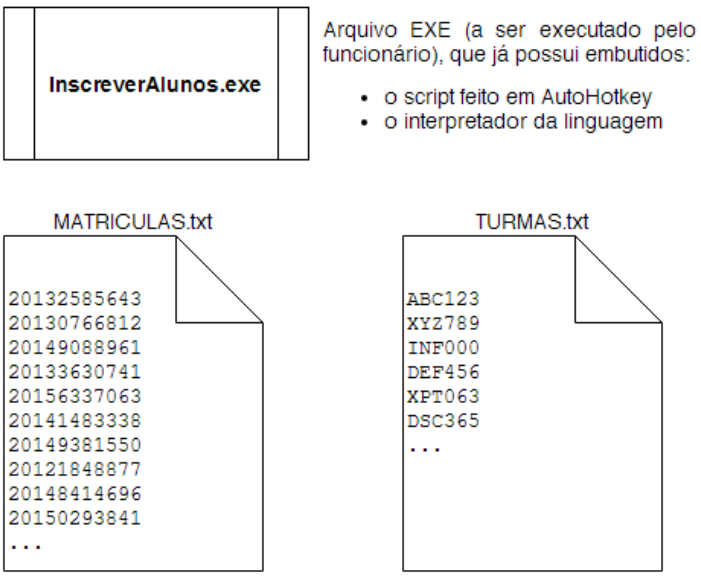


Figura 15: Arquivos necessários para o uso da solução desenvolvida

4 Conclusão

4.1 Considerações Finais

Ao final de nossa automação desenvolvida para a secretaria do curso, obtivemos uma avaliação positiva de todos os funcionários que participaram dos testes realizados ao longo do desenvolvimento. Além de tornar o processo mais rápido fazendo em 20 minutos as mesmas ações que manualmente demandariam 45 minutos ou mais, aprimoramos também sua confiabilidade, pois atividades dessa natureza estão sempre sujeitas a um erro por parte do usuário. Minimizando a necessidade de interação humana, reduzimos também a quantidade de erros deste tipo que poderiam ocorrer.

Ainda assim, o benefício mais notável para os envolvidos foi a redução do tempo. Cabe aqui observar que o ganho proporcionado pela automação não é apenas a diferença entre estes valores (25 minutos). Na realidade, os 20 minutos nos quais a máquina realiza a tarefa dispensam esforço manual, permitindo que o usuário realize outras atividades paralelamente. Neste caso, pode-se aproveitar este período de tempo para organizar documentos físicos e atender alunos, entre outras possibilidades.

Apesar de não terem acompanhado a codificação do script, os funcionários ficaram curiosos ao observar a evolução do processo automatizado ao longo do tempo e receberam explicações sobre alguns breves trechos de código. Isso possibilitou que avaliássemos (ainda que informalmente) suas opiniões sobre o AutoHotkey. Depois de presenciar o software efetuando rapidamente a inscrição dos alunos, os funcionários relataram que desejam aprender mais sobre a ferramenta no futuro, pois consideram que não teriam grandes dificuldades em aprender os conceitos básicos da linguagem e que este seria um conhecimento interessante e muito útil para suas tarefas rotineiras – tanto em seus computadores pessoais como nos computadores de trabalho.

4.2 Limitações e Dificuldades

As greves ocorridas em 2015 [130-133] reduziram a disponibilidade dos funcionários da secretaria, impactando diretamente na realização do estudo de caso deste trabalho. Consequentemente, foi necessário abandonar a ideia inicial de lecionar devidamente aos funcionários sobre o AutoHotkey, o que permitiria que os mesmos acompanhassem mais ativamente o desenvolvimento do script e que fosse possível avaliar o aprendizado ao final de todo o processo.

Outra dificuldade relacionada à escassez de tempo mencionada acima foi a questão do acesso ao SIE. Este acesso precisa ser feito nos computadores da secretaria e exige uma autenticação por usuário e senha, que não podem ser compartilhados por questões de segurança. Assim, não era possível dar continuidade ao desenvolvimento do script fora do horário de funcionamento da secretaria. Além disso, em pelo menos duas visitas à universidade, o sistema encontrou-se fora do ar durante todo o expediente, impossibilitando qualquer tipo de teste ou aprimoramento do código fonte.

4.3 Trabalhos Futuros

Como observamos neste trabalho, outros softwares para a realização de automações são tipicamente considerados úteis em cenários mais específicos, como o Sikuli para tarefas guiadas visualmente e o Selenium para interações com navegadores. A utilização destes não necessariamente substitui o AutoHotkey devido às diferenças de funcionalidades, mas podem ser mais intuitivas em certos casos. Sugerimos então uma possível avaliação do desempenho de tais ferramentas em conjunto com o AutoHotkey, delegando etapas distintas de um mesmo processo de acordo com a especialidade de cada programa.

Esperamos por fim que este trabalho contribua para a construção de uma base teórica, permitindo que mais artigos acadêmicos explorem o AutoHotkey e a questão da automação de tarefas na rotina de usuários finais. Certamente ainda existem muitas atividades executadas na própria universidade (em outros setores e cursos) que podem ser aprimoradas através de softwares de automação. Sugerimos então que o assunto continue a ser explorado, a fim de manter as informações atualizadas e observar as novas tendências deste tipo de ferramenta.

Referências Bibliográficas

[1] Parasuraman, R.; Riley, V. "*Humans and automation: Use, misuse, disuse, abuse*", Human Factors: The Journal of the Human Factors and Ergonomics Society, 1997; 39(2): 230-253

[2] C. Neves; L. Duarte; N. Viana; V. Ferreira. "*Os dez maiores desafios da automação industrial: as perspectivas para o futuro*", II Congresso de Pesquisa e Inovação da Rede Norte Nordeste de Educaçao Tecnológica, João Pessoa, Paraíba, Brasil, 2007

[3] Autor, D. H. "*Why Are There Still So Many Jobs? The History and Future of Workplace Automation*", The Journal of Economic Perspectives, 2015; 29(3): 3-30

[4] Abbink, D. A.; Mulder, M.; Boer, E. R. "*Haptic shared control: smoothly shifting control authority?*", Cognition, Technology & Work, 2012; 14(1): 19-28

[5] Zisman, M. D. "*Office automation: Revolution or evolution?*", 1978.

[6] Nicholl, A. O.; Bouberi Filho, J. J. "*Ambiente que Promove a Inclusão: Conceitos de Acessibilidade e Usabilidade*", Assentamentos Humanos Magazine, 2001; 3(2)

[7] "*GNU Accessibility Statement*", GNU Project, 2010-2015. Disponível em: <https://www.gnu.org/accessibility/accessibility.html>. Acesso em: 13 de Dezembro de 2015.

[8] Betke, M.; Gips, J.; Fleming, P. "The camera mouse: visual tracking of body features to provide computer access for people with severe disabilities", Neural Systems and Rehabilitation Engineering, IEEE Transactions, 2002; 10(1): 1-10

[9] Barbulescu, A. "*Software Products Accessibility Evaluation Metrics*", Revista Informatica Economica, 2008; 2(46): 86

[10] Burgstahler, S. "*Designing software that is accessible to individuals with disabilities*", Seattle, Washington: DO-IT, University of Washington-Seattle, 2002.

[11] Dubey, S. K.; Gulati, A.; Rana, A. "*Integrated model for software usability*", International Journal on Computer Science and Engineering, 2012; 4(3): 429-437

[12] Ohira, L. M. "*Identificação de requisitos para usabilidade de software assistivo*", Universidade Federal do Paraná, 2009.

[13] Erdogmus, H. "*CASCON'98 Workshop Report: Work injuries of computer users*", 1999.

[14] Kostaras, N.; Xenos, M. "*A study on how usability flaws in GUI design increase mouse movements and consequently may affect users' health*", Behaviour & Information Technology, 2011; 30(3): 425-436

[15] Liu, Z; Douglas, I. "Global Usability", Springer, 2011; ISBN 978-0-85729-303-9 (Print) / 978-0-85729-304-6 (Online)

[16] Scheiber, F.; Wruk, D.; Oberg, A.; Britsch, J.; Woywode, M.; Maedche, A.; Kahrau, F.; Meth, H.; Wallach, D.; Plach, M. "*Software Usability in Small and Medium Sized Enterprises in Germany: An Empirical Study*", In: Maedche, A.; Botzenhardt, A.; Neer, L. "*Software for People*", 2012; 39-52; ISBN 978-3-642-31370-7 (Print) / 978-3-642-31371-4 (Online)

[17] Kaplan, S.; Reeser, T.; Kelly, F. "*Citrix MetaFrame for Windows Server 2003: The Official Guide*". New York: McGraw-Hill, 2003; ISBN 0-07-219566-5.

[18] McHenry, K.; Bajcsy, P. "*Framework Converts Files of Any Format*", Society of Photo-Optical Instrumentation Engineers (SPIE) Newsroom, 2009

[19] McHenry, K.; Kooper, R.; Bajcsy, P. "*Taking Matters into Your Own Hands: Imposing Code Reusability for Universal File Format Conversion*", Microsoft eScience Workshop, 2009

[20] McHenry, K.; Kooper, R.; Bajcsy, P. "*Towards a Universal, Quantifiable, and Scalable File Format Converter*", The IEEE International Conference on eScience, 2009

[21] Lee, Y. H.; Song, H. T.; Suh, J. S. "*Quantitative computed tomography (QCT) as a radiology reporting tool by using optical character recognition (OCR) and macro program*", Journal of digital imaging, 2012; 25(6): 815-818

[22] Lim, J. H.; Yang, H. J.; Jung, K. H.; Yoo, S. C.; Paek, N. C. "*Quantitative trait locus mapping and candidate gene analysis for plant architecture traits using whole genome re-sequencing in rice*", Molecules and Cells, 2014; 37(2): 149-160

[23] Engimann, J.; Santarelli, T.; Zachary, W.; Hu, X.; Cai, Z.; Mall, H.; Goldberg, B. S. "*Game-based Architecture for Mentor-Enhanced Training Environments (GAMETE)*". In: "*Generalized Intelligent Framework for Tutoring (GIFT) Users Symposium (GIFTSym2)*", 2015

[24] Velasco, F. J.; Revestido, E.; Lastra, F. J.; Riola, J. M.; Díaz, J. J. "*Parameter Estimation and Control of an Unmanned Underwater Vehicle*", Journal of Maritime Research, 2013; 10(3): 29-36

[25] Casas, P.; Schatz, R. "*Quality of Experience in Cloud services: Survey and measurements*", Computer Networks, 2014; 68: 149-165

[26] Borgersen, R. "*Getting More Productive with AutoHotKey*", University Teaching Services, 2012; 21(1): 24-25. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.259.2761&rep=rep1&type=pdf>. Acesso em: 13 de Dezembro de 2015.

[27] Lorenz, M.; Ovchinnikova, O. S.; Van Berkel, G. J. "*Fully automated laser ablation liquid capture surface analysis using nanoelectrospray ionization mass spectrometry*", Rapid Communications in Mass Spectrometry, 2014; 28(11): 1312-1320

[28] Colavito, K. W.; Madenci, E. "*Adhesive failure in hybrid double cantilever beams by digital image correlation*", 51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 2010.

[29] Balkman, J. D.; Siegel, A. H. "*Autopage and the Use of Computer Scripts to Automate Microtasks*", Journal of digital imaging, 2014; 27(4), 474-478

[30] 朱玉强(Zhu Yujiang). "*The Application of AutoHotkey to Real-time Consultation*", Library of Shandong Normal University, 2009. Disponível em: <http://www.sxlib.org.cn/xuehui/xhcbw/ddtsg/2009/1/201010/t20101013_102621.htm>. Acesso em: 13 de Dezembro de 2015.

[31] Liu, Y. K. "*Design And Implementation Of SAP Plug-in Based On Autohotkey*", Tese de Mestrado, Beijing University of Posts and Telecommunications, 2011. Disponível em: <http://globethesis.com/?t=2218330338953044>. Acesso em: 13 de Dezembro de 2015.

[32] Yi, Z.; Lingcang, C.; Yan, B. "*Programming Realization of Automatic Repeat Signals Acquisition Based on General Purpose Oscilloscope*", Chinese Journal of Scientific Instrument, 2008; 29(4): 487-490; ISSN 0254-3087. Disponível em: <http://www.edatop.com/down/paper/osc/%E7%A4%BA%E6%B3%A2%E5%99%A8-712ah434pgtqwp0.pdf>. Acesso em: 13 de Dezembro de 2015.

[33] 刘景云(Liu Jingyun). "*拒绝乌龙关机 为Windows 巧设关机"密码锁*" ", Computer Programming Skills & Maintenance, 2014; (19): 82-83; ISSN 1006-4052. Disponível em: <http://caod.oriprobe.com/articles/42883326/ju_jue_wu_long_guan_ji___wei_windows_qiao_she_guan.htm>. Acesso em: 13 de Dezembro de 2015.

[34] 鬼泣 (Gui Qi). "*AutoHotKey 让特定用户按键失效*", Computer Fan, 2015; (10): 57-57; ISSN 1005-0043. Disponível em: <http://caod.oriprobe.com/articles/44733913/autohotkey_rang_te_ding_yong_hu_an_jian_shi_xiao_.htm>. Acesso em: 13 de Dezembro de 2015.

[35] Dunning, J. "*A Beginner's Guide to AutoHotkey*", ComputorEdge E-Books, 2014; 3. Disponível em: <http://www.computoredgebooks.com/A-Beginners-Guide-to-AutoHotkey-All-File-Formats_c29.htm>. Acesso em: 13 de Dezembro de 2015.

[36] Dunning, J. "*Digging Deeper into AutoHotkey*", ComputorEdge E-Books, 2013. Disponível em: <http://www.computoredgebooks.com/Digging-Deeper-into-AutoHotkey-All-File-Formats_c30.htm>. Acesso em: 13 de Dezembro de 2015.

[37] Dunning, J. "*AutoHotkey Applications*", ComputorEdge E-Books, 2014. Disponível em: <http://www.computoredgebooks.com/AutoHotkey-Applications-All-File-Formats_c31.htm>. Acesso em: 13 de Dezembro de 2015.

[38] Dunning, J. "*A Beginner's Guide to Using Regular Expressions in AutoHotkey*", ComputorEdge E-Books, 2015. Disponível em: <http://www.computoredgebooks.com/Regular-Expressions-in-AutoHotkey-EPUB-MOBI-and-PDF-Bundle-AUTOHOTKEY-5-BUNDLE.htm>. Acesso em: 13 de Dezembro de 2015.

[39] Santos, A. L. D. "*Controle de banda em redes TCP/IP utilizando o Linux*", Universidade Federal de Lavras, 2015

[40] Rohde, T. M. "*Desenvolvimento de um drive virtual: mouse e teclado*", Universidade Regional do Noroeste do Estado do Rio Grande do Sul, 2015

[41] Silva, L., Oliveira, L. C. "*Padrões de Auto Hot-Key (AHK) em Jogos Online*", 64ª Reunião Anual da SBPC, 2012

[42] "*Usage share of operating systems*", Wikipedia, 2007-2015.
Disponível em: <https://en.wikipedia.org/wiki/Usage_share_of_operating_systems>.
Acesso em: 13 de Dezembro de 2015.

[43] "*Browser, OS, Search Engine including Mobile Usage Share*", StatCounter Global Stats.
Disponível em: <http://gs.statcounter.com/>.
Acesso em: 13 de Dezembro de 2015.

[44] Calisir, F.; Calisir, F. "*The relation of interface usability characteristics, perceived usefulness, and perceived ease of use to end-user satisfaction with enterprise resource planning (ERP) systems*", Computers in human behavior, 2004; 20(4): 505-515.

[45] Czerwinski, M.; Horvitz, E.; Cutrell, E. "*Subjective duration assessment: An implicit probe for software usability*", In: "*Proceedings of IHM-HCI 2001 conference*", 2001; 19: 167-170

[46] Igbaria, M.; Nachman, S. A. "*Correlates of user satisfaction with end user computing: an exploratory study*", Information & Management, 1990; 19(2): 73-82

[47] "*AutoHotkey*", autohotkey.com.
Disponível em: <https://autohotkey.com/>.
Acesso em: 13 de Dezembro de 2015.

[48] "*AutoHotkey*", ahkscript.org.
Disponível em: <http://ahkscript.org/>.
Acesso em: 13 de Dezembro de 2015.

[49] Mallet, C. "*Re: Where did you hear about Autohotkey?*", AutoHotkey Community, 15 de Abril de 2005.
Disponível em: <https://autohotkey.com/board/topic/2950-where-did-you-hear-about-autohotkey/?p=19417>.
Acesso em: 13 de Dezembro de 2015.

[50] "*An AutoIt / AutoHotkey Comparison*", Dee Newcum, 9 de Maio de 2008.
Disponível em:
<http://paperlined.org/apps/autohotkey/autoit_and_autohotkey.html>.
Acesso em: 13 de Dezembro de 2015.

[51] "*Autohotkey.com Whois Lookup*", Who.is.
Disponível em: <https://who.is/whois/autohotkey.com>.
Acesso em: 13 de Dezembro de 2015.

[52] "*Database Access*", AutoHotkey Community, 5 de Março de 2004.
Disponível em: <https://autohotkey.com/board/topic/7-database-access>.
Acesso em: 13 de Dezembro de 2015.

[53] "*Autohotkey.net Whois Lookup*", Who.is.
Disponível em: <https://who.is/whois/autohotkey.net>.
Acesso em: 13 de Dezembro de 2015.

[54] Ahmed, A. "*AutoHotkey.net Status*", AutoHotkey Community, 25 de Junho de 2013.
Disponível em: <https://autohotkey.com/board/topic/94772-autohotkeynet-status/>.
Acesso em: 13 de Dezembro de 2015.

[55] Sadun, E. "*Download of the Day: AutoHotkey*", Lifehacker, 19 de Agosto de 2005.
Disponível em: <http://lifehacker.com/118270/download-of-the-day-autohotkey>.
Acesso em: 13 de Dezembro de 2015.

[56] Pash, A. "*Turn Any Action Into a Keyboard Shortcut: A Beginner's Guide to AutoHotkey*", Lifehacker, 30 de Outubro de 2007.
Disponível em: <http://lifehacker.com/316589/turn-any-action-into-a-keyboard-shortcut>.
Acesso em: 13 de Dezembro de 2015.

[57] Nanz, S.; Furia, C. A. "*A comparative study of programming languages in Rosetta Code*". Proceedings of the 37th International Conference on Software Engineering, 2015.

[58] Glenn, W. "*Show Us Your Best AutoHotkey Script*", Lifehacker, 16 de Maio de 2013.
Disponível em: <http://lifehacker.com/show-us-your-best-autohotkey-script-507227185>.
Acesso em: 13 de Dezembro de 2015.

[59] Mallet, C. "*My status and website changes*", AutoHotkey Community, 10 de Outubro de 2010.
Disponível em: <https://autohotkey.com/board/topic/58864-my-status-and-website-changes/>.
Acesso em: 13 de Dezembro de 2015.

[60] "*The AutoHotkey Foundation: Our History*", ahkscript.org.
Disponível em: <http://ahkscript.org/foundation/history.html>.
Acesso em: 13 de Dezembro de 2015.

[61] "*Scripts*", autohotkey.com.
Disponível em: <https://autohotkey.com/docs/Scripts.htm>.
Acesso em: 13 de Dezembro de 2015.

[62] "*AutoHotkey Downloads*", autohotkey.com.
Disponível em: <https://autohotkey.com/download/>.
Acesso em: 13 de Dezembro de 2015.

[63] "*AutoHotkey Beginner Tutorial*", autohotkey.com.
Disponível em: <https://autohotkey.com/docs/Tutorial.htm>.
Acesso em: 13 de Dezembro de 2015.

[64] Mallet, C. "*Re: Three thumbs up!*", AutoHotkey Community, 9 de Abril de 2005.
Disponível em: <https://autohotkey.com/board/topic/2916-three-thumbs-up/?p=18972>.
Acesso em: 13 de Dezembro de 2015.

[65] Mallet, C. "*Re: An update for AutoScriptWriter*", AutoHotkey Community, 27 de Agosto de 2005.
Disponível em: <https://autohotkey.com/board/topic/4839-an-update-for-autoscriptwriter/?p=29715>.
Acesso em: 13 de Dezembro de 2015.

[66] "*Add AU3_Spy.exe to the .zip downloads*", AutoHotkey Community, 13 de Fevereiro de 2015.

Disponível em: <https://www.autohotkey.com/boards/viewtopic.php?t=6402>.
Acesso em: 13 de Dezembro de 2015.

[67] "*SciTE4AutoHotkey*", fincs' AutoHotkey website.
Disponível em: <http://fincs.ahk4.net/scite4ahk/>.
Acesso em: 13 de Dezembro de 2015.

[68] "*GUI Creator (formerly Basic GUI Creator)*", AutoHotkey Community, 16 de Outubro de 2013.
Disponível em: <https://autohotkey.com/boards/viewtopic.php?t=303>.
Acesso em: 13 de Dezembro de 2015.

[69] Batista, R. U. "*Macro Creator v4.1.3 - Automation Tool (Recorder & Writer)*", AutoHotkey Community, 4 de Outubro de 2013.
Disponível em: <https://autohotkey.com/boards/viewtopic.php?t=143>.
Acesso em: 13 de Dezembro de 2015.

[70] "*Pulover's Macro Creator*", GitHub.
Disponível em: <https://github.com/Pulover/PuloversMacroCreator>.
Acesso em: 13 de Dezembro de 2015.

[71] " *iWB2 Learner (iWebBrowser2)*", AutoHotkey Community, 31 de Agosto de 2012.
Disponível em: <https://autohotkey.com/board/topic/84258-iwb2-learner-iwebbrowser2/>.
Acesso em: 13 de Dezembro de 2015.

[72] "*Windows API*", Microsoft Developer Network (MSDN), 25 de Março de 2010.
Disponível em: <https://msdn.microsoft.com/en-us/library/cc433218.aspx>.
Acesso em: 13 de Dezembro de 2015.

[73] "*What programming language does AutoHotKey use?*", AutoHotkey Community, 19 de Maio de 2006.
Disponível em: <https://autohotkey.com/board/topic/9038-what-programming-language-does-autohotkey-use/>.
Acesso em: 13 de Dezembro de 2015.

[74] "*AutoHotkey_L*", GitHub.
Disponível em: <https://github.com/Lexikos/AutoHotkey_L>.
Acesso em: 13 de Dezembro de 2015.

[75] Fung, W. J. "*A Predictive Text Completion Software in Python*", Python Papers Monograph, 2010; 2

[76] "*Threads*", autohotkey.com.
Disponível em: <https://www.autohotkey.com/docs/misc/Threads.htm>.
Acesso em: 13 de Dezembro de 2015.

[77] "*AutoHotkey_H*", autohotkey.net.
Disponível em: <http://www.autohotkey.net/~HotKeyIt/AutoHotkey/>.
Acesso em: 13 de Dezembro de 2015.

[78] "*AutoHotkey_H*", GitHub Pages.
Disponível em: <https://hotkeyit.github.io/v2/>.
Acesso em: 13 de Dezembro de 2015.

[79] "*Variables and Expressions*", autohotkey.com.
Disponível em: <https://www.autohotkey.com/docs/Variables.htm>.
Acesso em: 13 de Dezembro de 2015.

[80] "*what language is AHK most similar to ??*", AutoHotkey Community, 20 de Abril de 2009.
Disponível em: <https://autohotkey.com/board/topic/39920-what-language-is-ahk-most-similar-to/?p=249597>.
Acesso em: 13 de Dezembro de 2015.

[81] Slack, J. M. "*System Testing on the Cheap*", In: "*2010 Information Systems Educators Conference Proceedings*", 2010.

[82] "*Comparison of programming languages (strings)*", Wikipedia, 2006-2015.
Disponível em:
<https://en.wikipedia.org/wiki/Comparison_of_programming_languages_(strings)>.
Acesso em: 13 de Dezembro de 2015.

[83] Dunning, J. "*AutoHotkey Versus AutoIt*", ComputorEdge Online.
Disponível em:
<http://www.computoredge.com/AutoHotkey/Compare_AutoHotkey_vs_AutoIt_Review.html>.
Acesso em: 13 de Dezembro de 2015.

[84] "*Hotkeys (Mouse, Joystick and Keyboard Shortcuts)*", autohotkey.com.
Disponível em: <https://autohotkey.com/docs/Hotkeys.htm>.
Acesso em: 13 de Dezembro de 2015.

[85] "*List of Keys, Mouse Buttons, and Joystick Controls*", autohotkey.com.
Disponível em: <https://autohotkey.com/docs/KeyList.htm>.
Acesso em: 13 de Dezembro de 2015.

[86] "*Remapping Keys and Buttons*", autohotkey.com.
Disponível em: <https://autohotkey.com/docs/misc/Remap.htm>.
Acesso em: 13 de Dezembro de 2015.

[87] "*Hotstrings and Auto-replace*", autohotkey.com.
Disponível em: <https://autohotkey.com/docs/Hotstrings.htm>.
Acesso em: 13 de Dezembro de 2015.

[88] "*Macro Recorder - pricing*", jitbjit.com.
Disponível em: <https://www.jitbit.com/macro-recorder/purchase/>.
Acesso em: 13 de Dezembro de 2015.

[89] "*Action(s)*", app.jbbres.com.
Disponível em: <http://app.jbbres.com/actions/>.
Acesso em: 13 de Dezembro de 2015.

[90] "*Selenium*", seleniumhq.org.
Disponível em: <http://seleniumhq.org>.
Acesso em: 13 de Dezembro de 2015.

[91] "*Selenium Tutorial: Project*", NewCircle.
Disponível em: <https://newcircle.com/bookshelf/selenium_tutorial/project>.
Acesso em: 13 de Dezembro de 2015.

[92] Besson, F. M.; Beder, D. M.; Chaim, M. L. "*Framework para Execução Automatizada de Testes de Aplicações Web*", Universidade de São Paulo, 2008.

[93] "*Script-it.exe (NT 4 Server)*", SS64.
Disponível em: <http://ss64.com/nt/scriptit.html>.
Acesso em: 13 de Dezembro de 2015.

[94] "*Script-it.exe*", narkive.com, 1 de Outubro de 2003.
Disponível em:
<http://microsoft.public.win32.programmer.tools.narkive.com/rC6Ve5j1/scriptit-exe>.
Acesso em: 13 de Dezembro de 2015.

[95] Mar-elia, D. "*ScriptIt*", Windows IT Pro, 30 de Novembro de 1998.
Disponível em: <http://windowsitpro.com/systems-management/scriptit>.
Acesso em: 13 de Dezembro de 2015.

[96] "*AutoIt*", autoitscript.com.
Disponível em: <https://www.autoitscript.com/>.
Acesso em: 13 de Dezembro de 2015.

[97] "*AutoIt and Developer History*", autoitscript.com.
Disponível em: <https://www.autoitscript.com/autoit3/docs/intro/dev_history.htm>.
Acesso em: 13 de Dezembro de 2015.

[98] "*AutoIt Changelog (including beta version changes)*", autoitscript.com.
Disponível em:
<https://www.autoitscript.com/autoit3/docs/autoit_changelog_complete.txt>.
Acesso em: 13 de Dezembro de 2015.

[99] Bennett, J. "*Licensing Opinions*", AutoIt Forums, 2 de Janeiro de 2005.
Disponível em: <https://www.autoitscript.com/forum/topic/7204-licensing-opinions/>.
Acesso em: 13 de Dezembro de 2015.

[100] "*New version of Autoit3*", AutoIt Forums, 20 de Julho de 2010.
Disponível em: <https://www.autoitscript.com/forum/topic/117399-new-version-of-autoit3/>.
Acesso em: 13 de Dezembro de 2015.

[101] "*AutoIt Downloads*", autoitscript.com.
Disponível em: <https://www.autoitscript.com/site/autoit/downloads/>.
Acesso em: 13 de Dezembro de 2015.

[102] "*AutoIt Script Editor*", autoitscript.com.
Disponível em: <https://www.autoitscript.com/site/autoit-script-editor/>.
Acesso em: 13 de Dezembro de 2015.

[103] "*Sikuli Script*", sikuli.org.
Disponível em: <http://www.sikuli.org/>.
Acesso em: 13 de Dezembro de 2015.

[104] "*Sikuli*", Launchpad.
Disponível em: <https://launchpad.net/sikuli>.
Acesso em: 13 de Dezembro de 2015.

[105] "*Sikuli*", GitHub.
Disponível em: <https://github.com/sikuli/sikuli>.
Acesso em: 13 de Dezembro de 2015.

[106] Hocke, R. "*SikuliX powered by RaiMan*", sikulix.com.
Disponível em: <http://www.sikulix.com/>.
Acesso em: 13 de Dezembro de 2015.

[107] "*SikuliX-2014 (version 1.1.x)*", GitHub.
Disponível em: <https://github.com/RaiMan/SikuliX-2014>.
Acesso em: 13 de Dezembro de 2015.

[108] "*Hello World (Windows) – Sikuli X 1.0 documentation*", sikuli.org.
Disponível em: <http://doc.sikuli.org/tutorials/helloworld/helloworld-win.html>.
Acesso em: 13 de Dezembro de 2015.

[109] Hocke, R. "*SikuliX QuickStart*", sikulix.com.
Disponível em: <http://www.sikulix.com/quickstart.html>.
Acesso em: 13 de Dezembro de 2015.

[110] "*Sikuli Documentation – Sikuli X 1.0 documentation*", sikuli.org.
Disponível em: <http://doc.sikuli.org/>.
Acesso em: 13 de Dezembro de 2015.

[111] Hocke, R. "*Sikuli / SikuliX Documentation for version 1.1+ (2014 and later)*", Read the Docs.
Disponível em: <http://doc.sikuli.org/>.
Acesso em: 13 de Dezembro de 2015.

[112] Lin, P. C.; Huang, S. K. "*A Regression Testing Framework for QEMU-based System and Software Development*", National Chiao Tung University, 2013

[113] "*AutoHotkey v2*", autohotkey.com.
Disponível em: <https://autohotkey.com/v2/>.
Acesso em: 13 de Dezembro de 2015.

[114] "*SikuliX2 (version 2.0.x)*", GitHub.
Disponível em: <https://github.com/RaiMan/SikuliX2>.
Acesso em: 13 de Dezembro de 2015.

[115] "*Sikuli 2.0.0 (SikuliX2)*", Launchpad.
Disponível em: <https://launchpad.net/sikuli/+milestone/2.0.0>.
Acesso em: 13 de Dezembro de 2015.

[116] "*AutoHotkey vs. AutoIt?*", AutoHotkey Community, 26 de Janeiro de 2006.
Disponível em: <https://autohotkey.com/board/topic/7019-autohotkey-vs-autoit/>.
Acesso em: 13 de Dezembro de 2015.

[117] Besiex, Q. von. "*AutoHotKey versus AutoIt*", quinxy.com, 29 de Maio de 2012.
Disponível em: <http://quinxy.com/technology/autohotkey-versus-autoit/>.
Acesso em: 13 de Dezembro de 2015.

[118] "*AutoIt or AutoHotkey?*", AdvancedCase (blogspot.com), 23 de Maio de 2013.
Disponível em: <http://advancedcase.blogspot.com/2013/05/autoit-or-autohotkey.html>.
Acesso em: 13 de Dezembro de 2015.

[119] "*Autohotkey vs AutoIt v3*", WikiVS, 2011-2015.
Disponível em: <https://www.wikivs.com/wiki/Autohotkey_vs_AutoIt_v3>.
Acesso em: 13 de Dezembro de 2015.

[120] "*Autohotkey vs. AutoIt*", Outliner Software, 1 de Setembro de 2012.
Disponível em: <http://www.outlinersoftware.com/topics/viewt/4317>.
Acesso em: 13 de Dezembro de 2015.

[121] Hocke, R. "*Listening to Global Hotkeys*", Read the Docs.
Disponível em: <https://sikulix-2014.readthedocs.org/en/latest/interaction.html#listening-to-global-hotkeys>.
Acesso em: 13 de Dezembro de 2015.

[122] "*Function HotKeySet*", autoitscript.com.
Disponível em: <https://www.autoitscript.com/autoit3/docs/functions/HotKeySet.htm>.
Acesso em: 13 de Dezembro de 2015.

[123] "*Hotkey*", autohotkey.com.
Disponível em: <https://www.autohotkey.com/docs/commands/Hotkey.htm>.
Acesso em: 13 de Dezembro de 2015.

[124] "*Keyboard remapping*", AutoIt Forums, 27 de Janeiro de 2010.
Disponível em: <https://www.autoitscript.com/forum/topic/109006-keyboard-remapping/>.
Acesso em: 13 de Dezembro de 2015.

[125] "*SIE – DTIC*", unirio.br.
Disponível em: <http://www2.unirio.br/dtic/sie>.
Acesso em: 13 de Dezembro de 2015.

[126] "*The WinTitle Parameter & the Last Found Window*", autohotkey.com.
Disponível em: <https://www.autohotkey.com/docs/misc/WinTitle.htm>.
Acesso em: 13 de Dezembro de 2015.

[127] Sako, T. "*The synthesis of the motion capture data*", Comenius University, 2007.

[128] Daniluk, A. "*ABC Delphi 6*", Helion, 2002; ISBN 83-7197-504-X

[129] Gómez, A. R.; Benítez, G. R.; Fernández, R. G. "*ECGLib 1.1: Biblioteca de Clases para el Desarrollo de Aplicaciones de Electrocardiografía con Delphi*", II Congreso Latinoamericano de Ingeniería Biomédica, 2001.

[130] "*CoordMode*", autohotkey.com.
Disponível em: <https://www.autohotkey.com/docs/commands/CoordMode.htm>.
Acesso em: 13 de Dezembro de 2015.

[131] "*Informes de Greves*", Asunirio - Associação dos Servidores Técnicos-Administrativos da Unirio.
Disponível em: <http://www.asunirio.org.br/informe_greves.aspx>.
Acesso em: 13 de Dezembro de 2015.

[132] "*Fim da Greve dos Técnico-Administrativos*", UNIRIO Em Greve (wordpress.com), 8 de Outubro de 2015.
Disponível em: <https://unirioemgreve.wordpress.com/2015/10/08/fim-da-greve-dos-tecnico-administrativos/>.
Acesso em: 13 de Dezembro de 2015.

[133] "*Greve no ensino público federal do Brasil em 2015*", Wikipedia, 2015.
Disponível em: <https://pt.wikipedia.org/wiki/Greve_no_ensino_p%C3%BAblico_federal_do_Brasil

_em_2015>.
Acesso em: 13 de Dezembro de 2015.

[134] "*Comunicado CNG Nº 46*", ANDES-SN, 11 de Outubro de 2015. Disponível em: <http://grevenasfederais.andes.org.br/2015/10/10/comunicado-cng-no-46-11-de-outubro-de-2015/>.
Acesso em: 13 de Dezembro de 2015.

Alves, D. D. "*Acessibilidade no desenvolvimento de software livre*", Dissertação de Mestrado, Faculdade de Computação (FACOM), Universidade Federal de Mato Grosso do Sul (UFMS), 2011.

Kasper, M.; Correll, N.; Yeh, T. H. "*Abstracting perception and manipulation in end-user robot programming using Sikuli*", In: "*Technologies for Practical Robot Applications (TePRA)*", 2014 IEEE International Conference, 2014.

Givens, P.; Chakarov, A.; Sankaranarayanan, S.; Yeh, T. "*Exploring the internal state of user interfaces by combining computer vision techniques with grammatical inference*", Proceedings of the 2013 International Conference on Software Engineering, 2013

Yeh, T.; Chang, T. H.; Miller, R. C. "*Sikuli: using GUI screenshots for search and automation*", Proceedings of the 22nd annual ACM symposium on User interface software and technology, 2009.

Chu, Z.; Gianvecchio, S.; Koehl, A.; Wang, H.; Jajodia, S. "*Blog or block: Detecting blog bots through behavioral biometrics*", Computer Networks, 2013; 57(3): 634-646

Goossen, W. T. "*Detailed Clinical Models: Representing Knowledge, Data and Semantics in Healthcare Information Technology*", Healthcare informatics research, 2014; 20(3): 163-172

Kaszubski, P. "*Encouraging students to concordance: Towards better integration with the writing course workflow*", Adam Mickiewicz University, 2010.

Lahti, L.; Kurhila, J. "*Low-cost portable text recognition and speech synthesis with generic laptop computer, digital camera and software*", Universal Access in Human-Computer Interaction, Ambient Interaction, 2007.

Jeschke, S.; Pfeiffer, O.; Vieritz, H. "*Integrated Accessibility Models of User Interfaces for IT and Automation Systems*", Proceedings of the International Conference on Technology Communication and Education, 2008.

"*Why I use (and love) AutoHotKey*", One Hour Programming, 17 de Julho de 2010. Disponível em: <http://www.onehourprogramming.com/blog/2010/7/17/why-i-use-and-love-autohotkey-ahk.html>.
Acesso em: 13 de Dezembro de 2015.

"*Using Window Spy*", AutoHotkey Community, 6 de Fevereiro de 2011. Disponível em: <https://autohotkey.com/board/topic/63365-using-window-spy/>.
Acesso em: 13 de Dezembro de 2015.

Ochoa, T. A.; Kelly, M. L.; Stuart, S.; Rogers-Adkinson, D. "*The impact of PBL technology on the preparation of teachers of English language learners*", Journal of Special Education Technology, 2004; 19: 35-43

Mirenda, P.; Turoldo, K.; McAvoy, C. "*The impact of word prediction software on the written output of students with physical disabilities*", Journal of Special Education Technology, 2006; 21(3): 5

"*AutoHotkey speed vs. C++ and others*", AutoHotkey Community, 13 de Maio de 2011. Disponível em: <https://autohotkey.com/board/topic/66836-autohotkey-speed-vs-c-and-others/>.
Acesso em: 13 de Dezembro de 2015.

Scott, T. "*The Greatest Bodging Tool: AutoHotkey*", In: "*The Art of the Bodge: How I Made The Emoji Keyboard*", Youtube, 24 de Setembro de 2015.
Disponível em: <https://youtu.be/lIFE7h3m40U?t=168>.
Acesso em: 13 de Dezembro de 2015.

Soares, M. R. D. S. "*Automatização de testes funcionais utilizando a ferramenta Sikuli*", Trabalho de Conclusão do Curso de Bacharelado em Ciência da Computação, Universidade Estadual Paulista, 2012.

"*Scripting language*", Wikipedia, 2001-2015.
Disponível em: <https://en.wikipedia.org/wiki/Scripting_language>.
Acesso em: 13 de Dezembro de 2015.

"*Macro (computer science)*", Wikipedia, 2002-2015.
Disponível em: <https://en.wikipedia.org/wiki/Macro_(computer_science)>.
Acesso em: 13 de Dezembro de 2015.

Apêndice 1: Código Fonte do Script Desenvolvido

```
#NoEnv
#KeyHistory, 0
ListLines, Off
SetBatchLines, -1
DetectHiddenWindows, Off
#SingleInstance, IGNORE
#WinActivateForce
CoordMode, Mouse, Client
SetTitleMatchMode, RegEx
SetTitleMatchMode, Slow
DetectHiddenText, On
SetWorkingDir, %A_ScriptDir%

;##########################################
;######### CONFIGURAÇÃO DE ARQUIVOS #########
;##########################################
cfg_ArquivoMatriculas    := "MATRICULAS.txt"
cfg_ArquivoTurmas    := "TURMAS.txt"
;##########################################

glob_MsgEntrada := "O script para cadastrar alunos em turmas esta prestes a
ser executado."
glob_MsgEntrada .= "`n" . "`n" . "Antes de prosseguir, certifique-se de que os
arquivos '" cfg_ArquivoMatriculas
glob_MsgEntrada .= "' e '" cfg_ArquivoTurmas "' estão presentes no mesmo local
do script e preenchidos devidamente."
glob_MsgEntrada .= "`n" . "`n" . "Para abortar o script, basta apertar ESC a
qualquer momento."
glob_MsgEntrada .= "`n" . "`n" . "Deseja executar o script agora?"
MsgBox(glob_MsgEntrada,4096+32+4)
IfMsgBox, No
    ExitApp

glob_AtualMatricula := ""
glob_AtualTurma := ""
glob_Start := A_TickCount
glob_Matriculas := ""
glob_Turmas := ""
glob_ContagemMatriculas := 0
glob_ContagemTurmas := 0

try {
    glob_Matriculas :=
CarregarArquivoDeMatriculas(A_WorkingDir,cfg_ArquivoMatriculas)
    glob_Turmas := CarregarArquivoDeTurmas(A_WorkingDir,cfg_ArquivoTurmas)
    Loop, Parse, glob_Matriculas, % "`n", % "`r"
    {
        glob_AtualMatricula := Trim(A_LoopField)
        if ( glob_AtualMatricula = "" )
            continue
        SelecionarAluno(glob_AtualMatricula)
        Loop, Parse, glob_Turmas, % "`n", % "`r"
```

```
        {
            glob_AtualTurma := Trim(A_LoopField)
            if ( glob_AtualTurma = "" )
                continue
            Sleep, 20
            SelecionarTurma(glob_AtualTurma)
            glob_HWND := AtivarTela("Matrícula por Aluno")
            GetClientSize(glob_HWND,glob_W,glob_H)
            ;; COORDENADAS (CLIENT) BOTAO SALVAR:   W-125, H-35
            ;; COORDENADAS (CLIENT) BOTAO CANCELAR: W-50 , H-35
            ; Graças ao comando "SetTitleMatchMode, Slow" no início do script,
            ; podemos verificar se a lista de disciplinas na tela principal
foi
            ; alterada depois do clique, indicando que a operação foi
executada
            ; com sucesso (no caso de SALVAR).
            glob_Texto01 := ObterTextoDaListaDeMatriculas()
            glob_Texto02 := glob_Texto01
            Clicar( glob_W-125 , glob_H-35 )
            While ( glob_Texto01 = glob_Texto02 ) {
                if ( A_Index >= 10 )
                {
                    ; Entretanto, a rotina de obter o texto da janela
principal não é
                    ; 100% confiável. Então, ao invés de assumir que ocorreu
um erro,
                    ; simplesmente paramos o loop, mas verificamos antes se
uma
                    ; mensagem de erro do SIE foi exibida.
                    VerificarTelaDeErro()
                    break
                }
                Sleep, 50 + ((A_Index-1)*10)
                glob_Texto02 := ObterTextoDaListaDeMatriculas()
            }
            ;FUNCAO_EXCLUIR() ;; <-- comentada: rotina apenas para testes
finais
            glob_ContagemTurmas += 1
        }
        glob_ContagemMatriculas += 1
        glob_AtualTurma := ""
    }
} catch glob_Excecao {
    Abortar(glob_Excecao,glob_AtualMatricula,glob_AtualTurma)
}

glob_AtualMatricula := ""
glob_End := A_TickCount

glob_TempoSegundos := (glob_End-glob_Start)/1000
glob_TempoMinutos := glob_TempoSegundos/60

glob_MsgSaida := "O script foi executado com sucesso!"
glob_MsgSaida .= "`n`nProcessadas:"
glob_MsgSaida .= "`n`t" glob_ContagemMatriculas " matrículas"
glob_MsgSaida .= "`n`t" glob_ContagemTurmas " turmas"
glob_MsgSaida .= "`n`nTempo:"
glob_MsgSaida .= "`n`t" glob_TempoSegundos " segundos"
glob_MsgSaida .= "`n`t" glob_TempoMinutos " minutos"
MsgBox(glob_MsgSaida,4096+64)

ExitApp

;=====================================================================

*$ESC::
    Abortar("Abortado pelo usuário.",glob_AtualMatricula,glob_AtualTurma)
ExitApp
```

```
;=====================================================================

Abortar(exc="",matricula="",turma="") {
    txt := ""
    if ( IsObject(exc) ) {
        txt .= "Exception:"
        txt .= "`n" . "Msg:`t" exc.Message
        txt .= "`n" . "What:`t" exc.What
        txt .= "`n" . "Extra:`t" exc.Extra
        txt .= "`n" . "Line:`t" exc.Line
    } else {
        txt .= exc
    }
    txt .= "`n"
    txt .= "`n" . "Matrícula: " matricula
    txt .= "`n" . "Turma: " turma
    if ( !matricula OR !turma ) {
        txt .= "`n"
        txt .= "`n" . "Por favor corrija o problema "
        txt .= "antes de executar novamente."
    } else {
        txt .= "`n"
        txt .= "`n" . "Execute os seguintes passos manualmente:"
        txt .= "`n"
        txt .= "`n" . "* Cadastre o aluno " matricula " na turma " turma
        txt .= "`n" . "* Cadastre-o também nas turmas que aparecem "
        txt .= "DEPOIS de " turma " no arquivo"
        txt .= "`n" . "* Remova do arquivo de matrículas o aluno " matricula
        txt .= "`n" . "* Remova também as matrículas que aparecem "
        txt .= "ANTES de " matricula " no arquivo"
        txt .= "`n" . "* Execute o script novamente"
    }
    MsgBox(txt,4096+16)
    ExitApp
}

;---------------------------------------------------------------------

VerificarTelaDeErro() {
    if ( WinExist("Atenção! ahk_class TfrCpErrorsVisual") )
        throw { message: "Tela de Erro do SIE detectada!", what: "Erro_SIE",
file: A_LineFile, line: A_LineNumber }
}

;---------------------------------------------------------------------

CarregarArquivoDeMatriculas(p_Pasta,p_Arquivo) {
    l_Caminho := p_Pasta "\" p_Arquivo
    try {
        FileRead, l_Conteudo, %l_Caminho%
        if ( ErrorLevel )
            throw { message: "ErrorLevel = " ErrorLevel, what: "FileRead",
file: A_LineFile, line: A_LineNumber }
        l_Conteudo := RegExReplace(l_Conteudo,  "m)#[^\r\n]*"   )
        l_Conteudo := RegExReplace(l_Conteudo,  "[^\d\r\n]" )
        l_Conteudo := RegExReplace(l_Conteudo,  "[\r\n]+","`n"  )
        l_Conteudo := RegExReplace(l_Conteudo,  "(^\s*|\s*$)"   )
        return Trim(l_Conteudo)
    } catch e {
        e.extra := l_Caminho
        throw e
    }
}

;---------------------------------------------------------------------

CarregarArquivoDeTurmas(p_Pasta,p_Arquivo) {
```

```
    l_Caminho := p_Pasta "\" p_Arquivo
    try {
        FileRead, l_Conteudo, %l_Caminho%
        if ( ErrorLevel )
            throw { message: "ErrorLevel = " ErrorLevel, what: "FileRead",
file: A_LineFile, line: A_LineNumber }
        l_Conteudo := RegExReplace(l_Conteudo,  "m)#[^\r\n]*"   )
        l_Conteudo := RegExReplace(l_Conteudo,  "[^\w\r\n]" ) ; lembrete: "\w"
inclui dígitos e "_"
        StringReplace, l_Conteudo, l_Conteudo, % "_", % "", All
        l_Conteudo := RegExReplace(l_Conteudo,  "[\r\n]+","`n"  )
        l_Conteudo := RegExReplace(l_Conteudo,  "(^\s*|\s*$)"   )
        StringUpper, l_Conteudo, l_Conteudo
        return Trim(l_Conteudo)
    } catch e {
        e.extra := l_Caminho
        throw e
    }
}

;------------------------------------------------------------------

AtivarTela(p_Titulo,p_Class="Tfr.*") {
    if ( p_Class )
        p_Titulo .= " ahk_class " p_Class
    try {
        VerificarTelaDeErro()
        WinWait, %p_Titulo%,, 4
        if ( ErrorLevel )
            throw { message: "ErrorLevel = " ErrorLevel, what: "FileRead",
file: A_LineFile, line: A_LineNumber }
        l_hwnd := WinExist(p_Titulo)
        WinActivate, ahk_id %l_hwnd%
        if ( ErrorLevel )
            throw { message: "ErrorLevel = " ErrorLevel, what: "FileRead",
file: A_LineFile, line: A_LineNumber }
        WinWaitActive, ahk_id %l_hwnd%,, 1
        if ( ErrorLevel )
            throw { message: "ErrorLevel = " ErrorLevel, what: "FileRead",
file: A_LineFile, line: A_LineNumber }
        Sleep, 10
    } catch e {
        e.extra := p_Titulo
        throw e
    }
    return l_hwnd
}

;------------------------------------------------------------------

EsperarTelaFechar(p_Titulo) {
    p_Titulo .= " ahk_class Tfr.*"
    try {
        WinWaitClose, %p_Titulo%,, 5
        if ( ErrorLevel )
            throw { message: "ErrorLevel = " ErrorLevel, what: "FileRead",
file: A_LineFile, line: A_LineNumber }
    } catch e {
        e.extra := p_Titulo
        throw e
    }
}

;------------------------------------------------------------------

Clicar(p_X,p_Y,p_Vezes=1) {
    try {
        VerificarTelaDeErro()
```

```
        BlockInput, On
        MouseClick, Left, %p_X%, %p_Y%, %p_Vezes%
        BlockInput, Off
    } catch e {
        BlockInput, Off
        e.extra := "(" p_X "," p_Y ") x" p_Vezes
        throw e
    }
}

;-----------------------------------------------------------------

SelecionarAluno(id_matricula) {
    ; Clicar na lupa do aluno e esperar a próxima tela
    AtivarTela("Matrícula por Aluno")
    err := 1
    While ( err )
    {
        VerificarTelaDeErro()
        Clicar(142,122)
        WinWait, % "Localizar Dados do Aluno ahk_class Tfr.*",, 0.5
        err := ErrorLevel
        if ( A_Index >= 10 )
        {
            eMsg := "Aguardando tela de Localizar Aluno"
            throw { message: eMsg, what: A_ThisFunc, file: A_LineFile, line:
A_LineNumber, extra: id_matricula }
        }
    }
    ; Digitar a matrícula do aluno e apertar ENTER
    AtivarTela("Localizar Dados do Aluno")
    Sendraw, %id_matricula%
    Sleep, 50
    Send, {ENTER}
    EsperarTelaFechar("Localizar Dados do Aluno")
}

;-----------------------------------------------------------------

SelecionarTurma(id_turma) {
    AtivarTela("Matrícula por Aluno")
    ; O botão "Novo" fica na coordenada 60,40.
    ; A lupa da turma fica na coordenada 214,254.
    ; Enquanto a janela "Localizar Turma" não existir,
    ; tentar clicar em "Novo" e na Lupa.
    err := 1
    While ( err )
    {
        VerificarTelaDeErro()
        BlockInput, On
        MouseClick, Left, 60, 40
        MouseClick, Left, 214, 254
        BlockInput, Off
        WinWait, % "Localizar Turma ahk_class Tfr.*",, 0.5
        err := ErrorLevel
        if ( A_Index >= 10 )
        {
            eMsg := "Aguardando tela de Localizar Turma"
            throw { message: eMsg, what: A_ThisFunc, file: A_LineFile, line:
A_LineNumber, extra: id_turma }
        }
    }
    ; Clicar no local onde digitamos o código da turma
    ; e garantir que não existe nada digitado
    SelecionarELimparBuscaDeTurma()
    ; Se necessário, digitar um código inválido para limpar o texto
    ; de "registros encontrados" e apertar ENTER
    texto_status := ObterStatusSemEspacosDaTelaDeTurmas()
```

```
    if InStr(texto_status,"registro(s)encontrado(s)")
    {
        Sendraw, xxxxx
        Sleep, 20
        Send, {ENTER}
        While ( !InStr(texto_status,"Nenhumregistroencontrado") ) {
            VerificarTelaDeErro()
            texto_status := ObterStatusSemEspacosDaTelaDeTurmas()
        }
    }
    SelecionarELimparBuscaDeTurma()
    Sendraw, %id_turma%
    Sleep, 10
    Send, {ENTER}
    While ( !InStr(texto_status,"registro(s)encontrado(s)") ) {
        Sleep, 50 + ((A_Index-1)*10)
        VerificarTelaDeErro()
        if ( A_Index >= 25 )
            throw { message: "Turma não encontrada", what: A_ThisFunc, line:
A_LineNumber, extra: id_turma }
        texto_status := ObterStatusSemEspacosDaTelaDeTurmas()
    }
    ;// Clicar duas vezes na primeira turma para seleciona-la
    AtivarTela("Localizar Turma")
    Clicar(30,200,2)
    EsperarTelaFechar("Localizar Turma")
}

;-------------------------------------------------------------------

SelecionarELimparBuscaDeTurma() {
    AtivarTela("Localizar Turma")
    Clicar(60,40,2)
    Sleep, 50
    Send, {END}
    Loop, 10
        Send, {BACKSPACE}
    Sleep, 10
}

;-------------------------------------------------------------------

ObterStatusSemEspacosDaTelaDeTurmas() {
    ControlGetText, l_Texto, % "TStatusBar1", % "Localizar Turma ahk_class
Tfr.*"
    StringReplace, l_Texto, l_Texto, % " ", % "", All
    return l_Texto
}

;-------------------------------------------------------------------

ObterTextoDaListaDeMatriculas() {
    WinGetText, l_Texto, % "Matrícula por Aluno ahk_class Tfr.*"
    l_Saida := ""
    Loop, Parse, l_Texto, % "`n", % "`r"
        if ( RegExMatch(A_LoopField,"^[A-Z]{3}\d+$") )
            l_Saida .= A_LoopField
    return l_Saida
}

;-------------------------------------------------------------------

MsgBox(p_Texto="",p_Opcoes=0,p_Titulo="",p_Timeout=0) {
    BlockInput, Off
    MsgBox, % p_Opcoes, % p_Titulo, % p_Texto, % p_Timeout
}

;-------------------------------------------------------------------
```

```
; autohotkey.com/board/topic/91733-command-to-get-gui-client-areas-
sizes/?p=578581
GetClientSize(hwnd, ByRef w, ByRef h)
{
    VarSetCapacity(rc, 16)
    DllCall("GetClientRect", "uint", hwnd, "uint", &rc)
    w := NumGet(rc, 8, "int")
    h := NumGet(rc, 12, "int")
}

;------------------------------------------------------------------

/*
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; função excluir (apenas para testes finais!)
FUNCAO_EXCLUIR() {
    AtivarTela("Matrícula por Aluno")
    Sleep, 600
    Clicar(333,355,2)
    Sleep, 100
    Clicar(150,45)
    Sleep, 10
    AtivarTela("Confirmação","TMensagemCPDForm")
    Sleep, 10
    Send, {ENTER}
    Sleep, 10
    AtivarTela("Informação","TMensagemCPDForm")
    Sleep, 10
    Send, {ENTER}
    Sleep, 10
}
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
*/
```