基于界面可视信息提取的软件互动集成①

Software Interactive Integration Based on UI Information Retrieval

张　金[1,2]　邬　晶[1]　陈卓宁[1,2]　(1.华中科技大学 机械科学与工程学院 湖北 武汉 430074；
2.武汉开目信息技术有限责任公司 湖北 武汉 430223)

摘　要： 不同时期、不同类型的企业信息化软件间往往缺少集成接口和已知格式的外部数据而难于集成。如人们不得不经常进行产品设计数据、工艺数据的批量导入导出、手工抄录等重复单调的工作，以保证信息互通。利用界面信息提取和模拟人工界面操作自动化完成这些单调重复性工作通常是可行的。本文提出一种基于屏幕显示内容辨识界面元素的方法，从而实现界面显示信息的分检、界面元素的定位，即可根据业务流程需要自动地操作界面，实现软件间的信息互传与互动。

关键词： 软件集成 信息提取 对象辨识 界面操作 互传与互动

1　引言

人们已经注意到，各类信息化软件特别是早期软件由于只注重单一领域的工作而忽略与其他软件的联系，形成了众多的“信息孤岛”。人们在这些孤立的系统环境下工作，大部分时间是在进行一些重复、枯燥的操作，如产品设计、工艺数据的批量导入导出，手工抄录等。这些工作具有很大的相似性和重复性，人工操作极易出错，效率低下，成本较高[1]。解决此类问题的通常办法是利用软件二次开发接口进行二次开发。但是多数早期软件并未提供二次开发接口，或者接口有限，不能满足企业的各种集成需要。利用界面自动化技术模仿人的操作流程[2]，实现软件间可视化互动集成技术已经引起人们的注意。这种方法不受二次开发接口的限制，理论上只要软件存在界面，就有可能通过外部界面实现软件的自动化操作，实现有界面软件间的互动集成。

界面自动化技术操作软件界面的步骤通常为：一.获取待操作对象的窗口句柄；二.以句柄为参数发送消息或调用 API 实施对象操作(必要时需跨进程操作)。这种基于句柄[3]的方法能较好地操作软件界面上的标准对象，但操作非标准对象存在困难。非标准对象一般由用户定义，其内部结构(可响应的消息、可使用的成员函数)尚不清楚，因此基于句柄方法的界面自动化技术可能无法对其进行操作。针对这种情况，本文提出一种非标准对象的识别与操作方法，即基于屏幕信息进行对象识别、基于鼠标键盘模拟进行对象操作，弥补了基于句柄的方法在操作非标准对象方面的不足，能有效扩充界面自动化技术的应用范围。

2　非标准对象识别与操作原理

2.1 非标准对象及其子对象难以操作的解决方案

把能直接用基于窗口句柄的界面自动化技术进行识别与操作的对象称为标准对象，如 VC6 的编辑框(CEdit 类)、树(CTreeCtrl)、列表框(CListBox)等，不能这样操作的对象统称为非标准对象。本文主要论述非标准对象的识别与操作。非标准对象可进一步分为基本对象和组合对象。基本对象不含子对象，如非标准按钮、编辑框、标签等；组合对象含一个以上的子对象，如非标准树、列表框、网格和类似 IE Web 页面的对象，组合对象的特征是只能用窗口枚举的方式得到其窗口句柄，但无法利用该句柄操作其内部成员(如树、列表框的子项，网格的单元格，Web 页面中的按钮、超链接等)。本文把对象的这些内部成员统称为子对象。

① 基金项目:国家科技支撑计划(2006BAF01A44);国家高技术研究发展计划(863)(2007AA040605)
收稿时间:2008-12-07

对非标准对象而言，可通过枚举窗口的方法获得其句柄，却不都能基于该句柄实施操作。一个非标准的编辑框，可能不响应 WM_SETTEXT 和 WM_GETTEXT 消息，导致无法设置和获取编辑框内容；一个非标准的树，由于它不是标准类，所以根本不知道它有哪些方法可以用来对其实施操作；又如类似 Web 页面的非标准对象，其内部包含各种对象如按钮、超链接、编辑框等，要操作它们也非常困难。

人工操作软件界面的流程可概括为：瞄准待操作对象所在区域，移动鼠标或光标到该区域上，进行鼠标或键盘操作。界面自动化技术可以通过模拟这个过程来绕过非标准对象编程操作遇到的困难。在该过程中，找到对象所在区域是关键，只要获得了对象区域，就可以用程序模拟鼠标键盘进行操作。对基本对象，获得其句柄后利用 GetWindowRect 得到其区域即可进行模拟操作。而对组合对象，在得到对象区域后，还要进一步获取其子对象区域方能进行操作，这里采用基于屏幕信息的方法获取子对象区域：已知待操作子对象的屏幕信息(如标识名)，可在对象区域进行扫描并取出屏幕信息，当扫描到某位置，取出的屏幕信息和待操作对象的屏幕信息一致时，即找到子对象区域。

2.2 对象识别与操作流程

将非标准对象的识别与操作方法和一般界面自动化技术结合，软件界面对象的识别与操作流程如图 1 所示。

对象识别：首先枚举界面对象句柄得到类名，如刚好和待操作对象类名一致且界面上仅存在一个该类对象，则直接进入操作阶段。如界面上存在两个以上的该类对象，则进一步利用坐标信息进行甄别，从而得到待操作对象的句柄。

对象操作：对标准对象，则以句柄为参数发送消息或调用 API 进行操作。对非标准对象，先获得其对象区域，如果是基本对象，直接定位鼠标到该区域进行模拟操作；如果是组合对象，则基于屏幕信息以一定策略搜索其子对象，最后定位鼠标进行模拟操作。个别组合对象子对象较多，部分子对象没有显示在对象区域内，必要时需要滚动对象滚动条，暴露出子对象以供识别。

3 非标准对象识别与操作关键技术

3.1 基于屏幕文字信息的子对象识别

屏幕取词技术已广泛应用于字典类工具，其基本原理是使得鼠标所指位置的一小块界面区域无效，若该区域刚好处于一文字串内部，则引起整个文字串重绘，在重绘过程中通过拦截 API 获取重绘的文字串内容，从而取出文字。

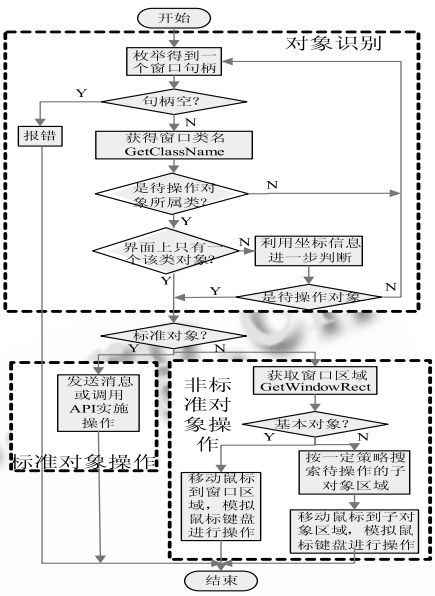

图 1　对象识别与操作流程

前文所述获得非标准基本对象的区域后可直接进行模拟操作，但对组合对象而言，还需要得到其子对象区域，组合对象的多数子对象都有一个唯一的标识名显示在子对象区域上，如树和列表框的子项标识名，Web 页面中的超链接、标签或按钮标识名。如果能在对象区域中找到子对象的标识名，则找到了子对象位置。

可以利用屏幕取词来协助搜索子对象位置。其策略为：在对象区域内取足够密度的点，依次使这些点附近一小块区域(如 3×3 像素)无效而重绘，在重绘时进行屏幕取词，当取出的文字和待操作子对象标识名匹配时，证明当前这块 3×3 像素的小区域位于待操作子对象上，于是进一步的鼠标键盘模拟操作成为可能。

3.1.1 屏幕取词交互过程

屏幕取词的交互过程如图 2 所示。

Windows 对窗体的绘制最终会调用到底层的五个 API 函数：TextOutA、TextOutW、ExtTextOutA、ExtTextOutW 和 BitBlt，传入这些函数的参数中，就包含了所要绘制的文字。因此，主控进程要获取屏幕上某窗体上某位置的文字，只需向该窗体所属的客户进程注入拦截代码并迫使该窗体在该位置重绘。在重绘过程中，注入的代码进行 API 拦截，获取重绘的文字内容并将文字传回控制进程即可，交互过程如图 2 所示。

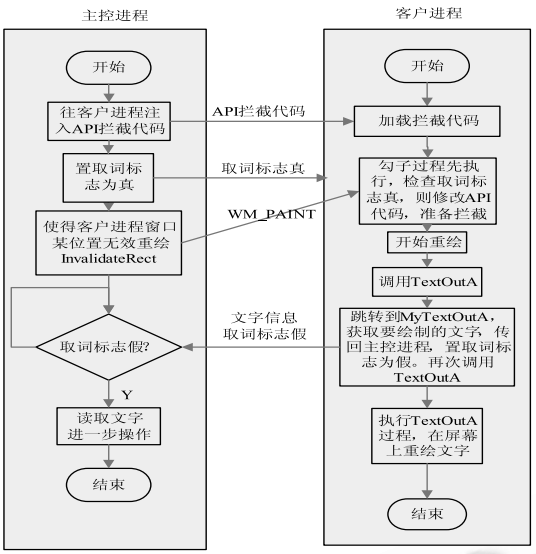

图 2　屏幕取词交互过程

3.1.2 屏幕取词关键实现描述

由图 2，屏幕取词的关键在于 API 拦截和跨进程数据传递。API 拦截的目的是为了获取重绘的文字信息，而跨进程数据传递一方面为了将客户进程中获取的文字信息传回主控进程中，另一方面是为了主控进程和客户进程共享取词标志而协同工作。

拦截 API 有两种方法，其一是修改模块输入节，其二是修改 API 代码页面。这里采用第二种方法来实现。实现描述如下。

1) 开发一个动态链接库组件，简称取词组件，将拦截 API 相关的代码写入取词组件中，另外再导出一个函数 SetAPIHook，这个函数用于为系统设置全局消息勾子，主控进程加载取词组件，调用 SetAPIHook，从而将拦截 API 的代码注入其他进程中。

2) 主控进程置取词标志为真并使客户进程窗口某区域无效(即发出 WM_PAINT 消息)，从而发起取词动作，客户进程响应 WM_PAINT 消息时，首先调用勾子过程函数，钩子过程函数定位到相关 API 的代码页面，把这些 API 代码的第一条指令替换成无条件跳转指令“JMP XXXX”，该指令跳转目的地的为取词代码的位置。执行完钩子函数后开始重绘。

3) 重绘过程调用相关 API 进行文字输出，由于 API 第一条指令已被修改为跳转指令，所以程序执行点跳转到取词代码，取词代码取出文字并置取词标志为假，然后恢复相关 API 的代码页面，重新调用这些 API 完成文字绘制。

4) 采用跨进程数据传递将数据传回主控进程。

由于上述过程是采用动态链接库方式注入拦截代码，因此采用公共节的方式进行跨进程数据传递和共享[4]。公共节里边声明字符串变量 g_strGet 用于保存取回的文字信息，声明取词标志 g_bGetWord 用于主控进程和客户进程同步。控制进程和客户进程可以直接对这两个变量进行读写，它们在所有进程中是共享的。相关实现代码如下：

```
#pragma data_seg(".Seg")
char g_strGet[MAXLEN] = "";
//保存取回的文字信息
BOOL bGetWord=FALSE;
//取词标志，用于主控进程和客户进程同步
#pragma data_seg()
#pragma comment(linker,"-.Seg: rws")
//share_seg 为公共节
```

3.2 基于模拟的非标准对象操作

通过基于屏幕信息的识别过程后，所有非标准(子)对象的区域已经获取到，可进行鼠标定位，然后模拟鼠标键盘操作。操作的过程高度模拟人工操作过程，无需句柄，无需发送消息，因此克服了一般界面自动化技术遇到的困难。

3.2.1 鼠标定位

对象识别过程提供了对象所在的区域。要操作对象，首先要将鼠标定位到该对象上，可用 SetCursorPos 来实现鼠标定位。如将鼠标定位到(XPos，YPos)的代码如下。

```
SetCursorPos(XPos,YPos);
```

3.2.2 鼠标、键盘模拟操作

鼠标定位到指定位置之后，根据不同的控件进行不同的操作。如果是编辑框控件，往往是先点击编辑框使其获得焦点，然后键盘输入数据；如果是树控件，往往是对其进行双击使其展开或修改其文字项目等。Windows 提供的 API SendInput 可以模拟鼠标键盘操作。SendInput 需要三个参数，其中最重要的是 INPUT 类型的参数，INPUT 是一个结构，包含一个类型为 DWORD 的 type 变量和一个联合类型的变量。Type 的取值有 INPUT_MOUSE、INPUT_KEYBOARD 和 INPUT_HARDWARE，分别代表当前模拟类型为鼠标、键盘和硬件操作。根据 type 的不同，联合类型变量相应可以取 MOUSEINPUT、KEYBDINPUT 和 ARDWAREINPUT，分别代表模拟的鼠标、键盘和硬件操作的具体内容。

以模拟鼠标双击为例，代码如下。

```
INPUT  input[4];
input[0].type=INPUT_MOUSE;
input[0].mi.dwFlags=MOUSEEVENTF_LEFTDOWN;
input[1].type=INPUT_MOUSE;
input[1].mi.dwFlags=MOUSEEVENTF_LEFTUP;
input[2]=input[0];
input[3]=input[1];
SendInput(4,input,sizeof(INPUT));
```

4　应用实例

开目公司内部所有日常工作都在信息化平台上进行，TestTrackPro(TTP)是通用缺陷管理工具。由项目需要，需把信息化平台的项目评审意见自动导出到 TTP 中进行集中管理，即实现信息化平台与 TTP 的集成。由于 TTP 缺乏二次开发接口，因此它们的集成只能采用界面自动化技术来实现。

以导出“ CATIA 轻量化浏览预研 ”项目评审意见为例，需求简述如下：启动信息化平台，依次展开如图 3 所示的树，在网格中找到要导出的项目，实施操作导出为中间文件，接下来启动 TTP，打开菜单，导入刚才的中间文件。要求自动实现这个流程。

图 3　信息化平台与 TTP 集成

采用基于句柄方式的界面自动化技术可以完成集成过程中的大多数操作，但对图 3 所示的非标准树和网格组合对象，则只能先基于屏幕信息识别其子对象，再基于鼠标键盘模拟实施操作。要导出“ CATIA 轻量化浏览预研 ”项目的评审意见，需要依次展开非标准树“ 开目公司-活动-技术中心开发任务 ”子对象，识别与操作过程如图 4 所示。

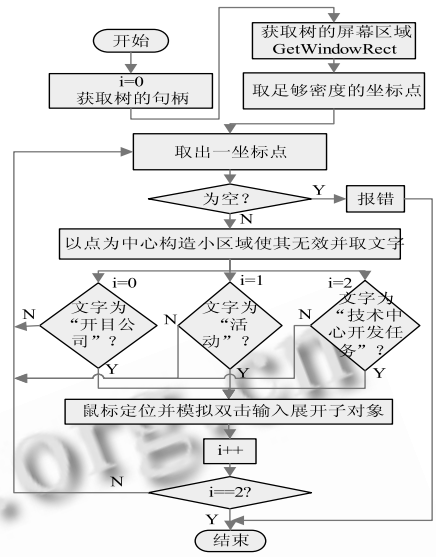

图 4　对象分类管理器树操作流程举例

为信息化平台和 TTP 集成而开发的界面自动操控软件中，将树和网格对象的操作封装起来，用户只需把所有要导出的项目及其在树对象中的路径编制成任务列表，操控软件在计算机空闲时读取任务列表并按预定流程自动进行操作，无需人工值守。

5　总结

提出一种基于屏幕信息提取与分析识别界面对象、模拟鼠标键盘操作的技术，解决一般界面自动化技术难于操作非标准对象及其子对象的问题。利用该技术，开发出了界面自动操控软件，模拟人在孤立系统下的工作流程，为一些企业解决了比较棘手的软件间的无缝集成问题，例如本文所举的信息化平台与 TTP 的集成用例。该技术是一种通用技术，不仅适用于行业内的软件集成，也适用于跨行业甚至跨学科的软件间集成。

参考文献

1 胡滨.软件自动测试工具的研究. 现代电子技术, 2007,28(18):97－99.
2 翟立东,孙丽萍.软件自动测试方法的研究与实现. 大连铁道学院学报, 2005,26(4):52－56.
3 于城蛟,刘更,王海伟.基于 Windows 消息的软件界面集成技术研究.计算机仿真, 2007,24(12):278－282.
4 Richter J,王建华,张焕生,侯丽坤.Windows 核心编程. 北京:机械工业出版社, 2000:515－564.