SAP2000API 二次开发-Python 示例

筑信达　孙雪艳

SAP2000 是通用结构分析与设计软件，在民用建筑、场馆结构、工业建筑中都有使用。使用软件进行模型处理的过程中，会遇到手工操作繁琐的过程，如格构构件建模、圆截面构筑物风荷载体型系数、钢结构防火设计等。这些问题在筑信达工具箱中均给出了解决方案，使用的是 SAP2000 提供的 API 函数。SAP2000 从 V11 版本开放 API（Application Programming Interface）接口。工程师通过 API 接口函数控制 SAP2000 软件的使用，进行建模、荷载设置、模型编辑、分析结果提取等，编制属于自己的工具。

API 函数可以使用多种编译环境进行开发，包括 VBA、VB、C#、Intel Visual Fortran、C++、MATLAB、Python 等。本文介绍使用 Python 语言进行 SAP2000API 二次开发的过程。

1 软件版本与 IDE 环境

Python 是一种解释型语言，关键字少，结构简单，易于学习。Python 既支持面向过程的函数编程也支持面向对象的抽象编程，具有丰富的库文件，可以进行各种类型项目的开发。

在 API 的帮助文件 CSI_OAPI_Documentation.chm 中建议使用 3.4.3 或更高级的版本 Python，本文作者使用的 Python 版本是 3.8.10。

编写 Python 代码的编译环境有多种，在安装 Python 后，自带 IDLE（Python's Integrated Development and Learning Environment）编辑器。该编辑器使用简单、通用，且支持不同设备。也可以安装其他类型的 IDE（Integrated Development Environment）进行代码编写与调试，本文作者使用的 IDE 为 PyCharm。PyCharm 带有一整套可以帮助用户在使用 Python 语言开发时提高效率的工具，比如调试、语法高亮、项目管理、代码跳转、智能提示、自动完成、单元测试、版本控制。

2 开发环境配置

在 Python 中通过 COM 组件调用 SAP20000 的接口进行软件的控制。使用 Python 不能直接调用 COM 组件，需要借助可以访问 COM 组件的库 comtypes。comtypes 是一个轻量级的 python 库，通过这个库文件，直接对 COM 组件进行对象创建，实现对 SAP2000 的调用。

第一次使用 comtypes 时，需要进行库文件的安装。安装库文件的方法有两种。一种是通过 pip 命令安装，一种是在 pycharm 中安装。两种方式安装后在 PyCharm 中均可以使用。

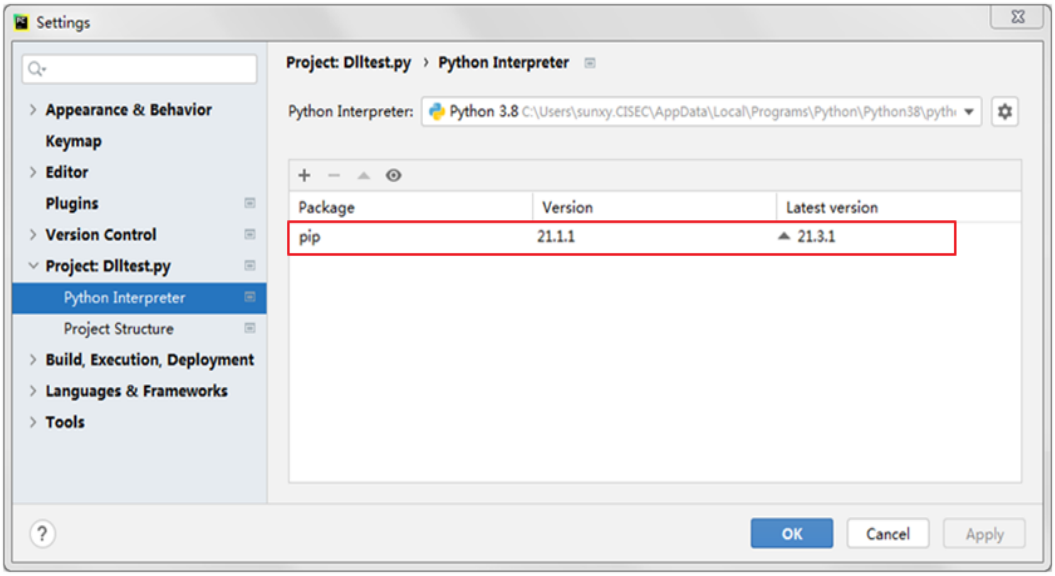

图 1 设置对话框

在设置窗口中，点击 Python Interpreter，设置 Python Interpreter 解释程序为系统的路径。图 1 红色圈中显示当前已经安装库文件。

2.1 Pip 命令安装

在开始菜单中调出 cmd 命令窗口，输入 python –m pip install comtypes 命令，自动联网进行 comtypes 库的安装。安装完成后，显示 Sucessfully installed contypes-1.1.10，表示 comtypes 库安装成功。

打开 PyCharm 的设置窗口，可以看到已经增加 comtypes 库文件。

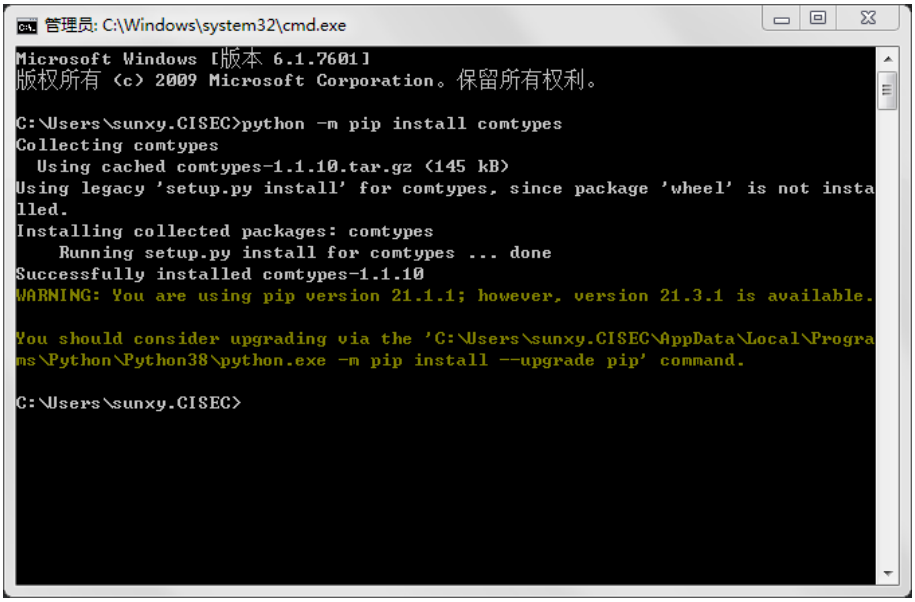

图 2 comtypes 库安装

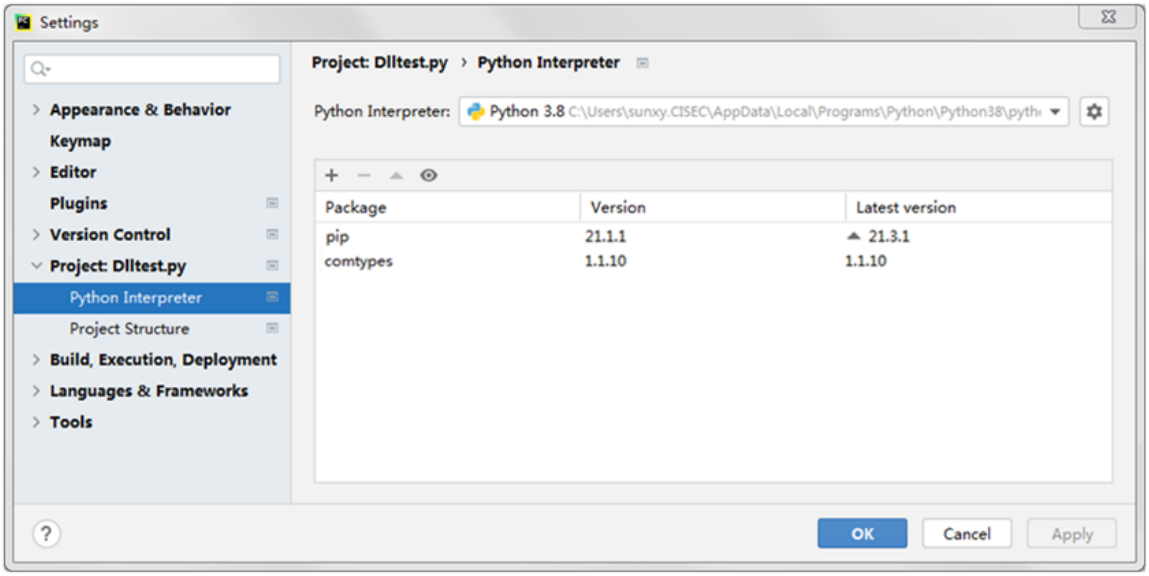

图 3 库文件显示

2.2 PyCharm 中安装

在 PyCharm 设置窗口中点击“+”按钮，在弹出的可用库文件窗口中，输入库名称查找库文件并安装，安装完成后，返回设置对话框，可以看到已经添加的 comtypes 库。

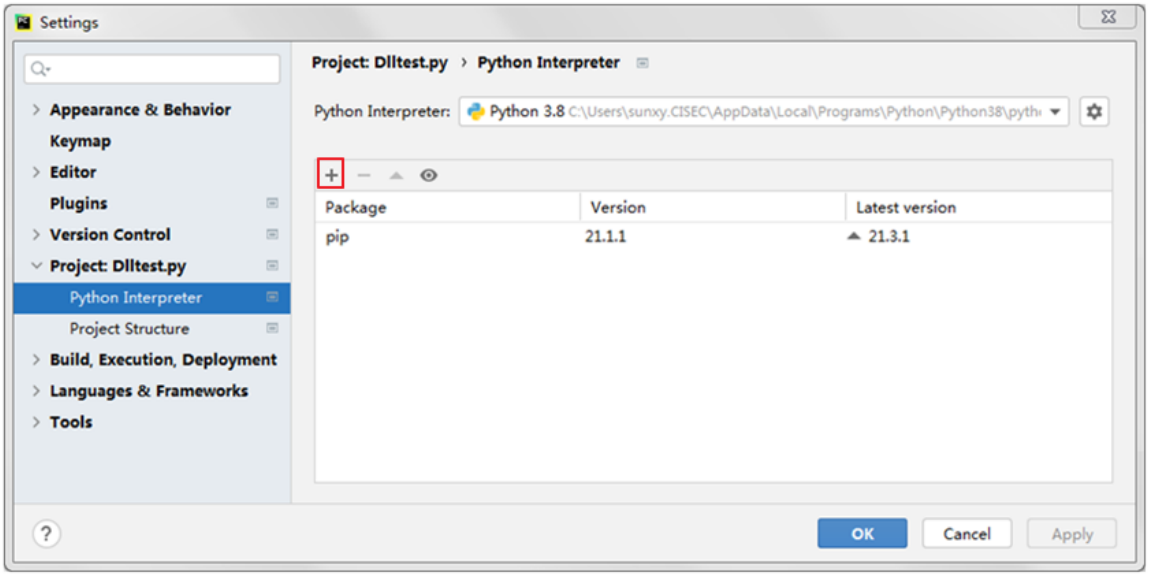

图 4 添加库文件

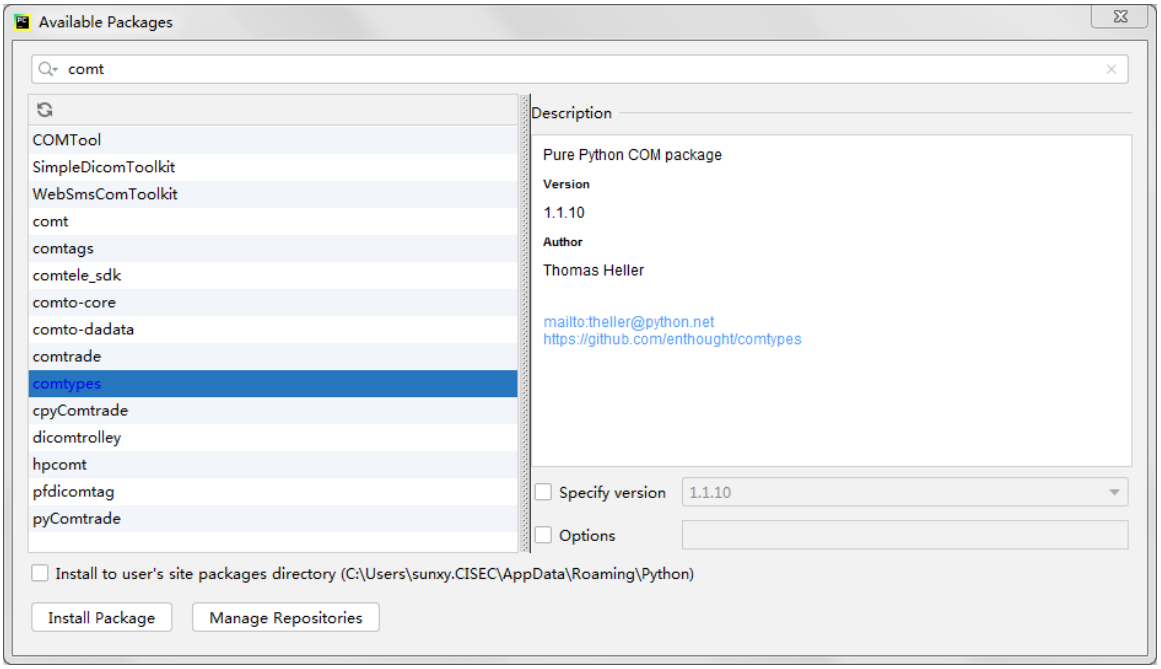

图 5 库文件查找与安装

3 开发示例

API 帮助文档 Example Code 中的 Example 7 是采用 COM 组件方式进行 SAP2000 API 开发的示例代码，介绍使用 Python 语言进行 SAP2000 API 二次开发的过程，包括调用 SAP2000 方式、建立模型、指定荷载、运行分析、提取结果等，本文以此为例介绍 Python 开发的过程。

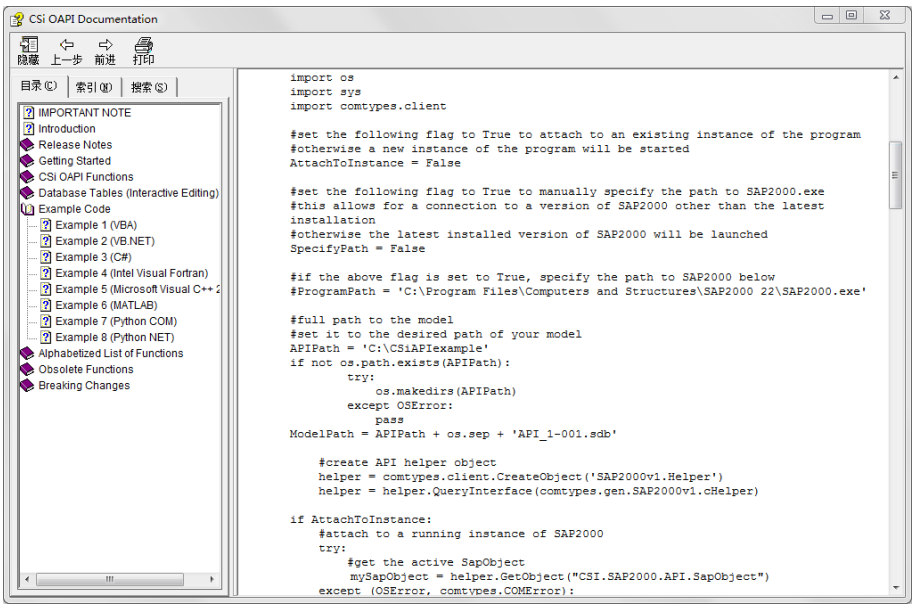

图 6 API 帮助文档 Python 示例代码

建立新的 py 文件，使用 import 导入相关的库，在本例中导入 sys 库和 comtypes 库。

```
import sys
import comtypes.client
```

图 7 导入库文件

3.1 启动 SAP2000 的两种方式

通过 comtypes 库提供的 CreateObject 创建 SAP2000 的实例，函数参数为所要创建的应用程序类名。先尝试打开已经运行的 SAP2000，如果没有，再打开新的 SAP2000，代码如图 8 所示。

```
helper = comtypes.client.CreateObject('SAP2000v1.Helper')
try:
    mySapObject = helper.GetObject("CSI.SAP2000.API.SapObject")
    if mySapObject==None:
        raise Exception("未获得当前运行的SAP2000。")
except(BaseException,comtypes.COMError):
    print("未获得当前运行的SAP2000。")
    ProgramPath = "D:\Program Files\Computers and Structures\SAP2000 23\SAP2000.exe"
    try:
        mySapObject = helper.CreateObject(ProgramPath)
    except (OSError, comtypes.COMError):
        print("不能通过路径打开SAP2000" + ProgramPath)
        try:
            # create an instance of the SAPObject from the latest installed SAP2000
            mySapObject = helper.CreateObjectProgID("CSI.SAP2000.API.SapObject")
        except (OSError, comtypes.COMError):
            print("不能通过ProgID打开SAP2000。")
            sys.exit(-1)
    mySapObject.ApplicationStart()
```

图 8 打开 SAP2000

1）附加到当前的 SAP2000

使用 helper.GetObject("CSI.SAP2000.API.SapObject")函数得到当前运行的 SAP2000 进程，函数参数为 programID，对于

SAP2000V23 版本，此参数为 CSI.SAP2000.API.SapObject，可以在帮助文件中获得这个值。

2）启动新的 SAP2000

根据 SAP2000.exe 文件的路径或者 SAP2000 的 program ID 来启动 SAP2000 软件。使用函数 helper.CreateObject(ProgramPath)，根据 exe 文件路径建立 SapObject 实例，参数为 exe 文件路径。也可以使用 helper.CreateObjectProgID("CSI.SAP2000.API.SapObject")函数根据软件的 program ID 启动建立 SapObject 实例。通过 mySapObject.ApplicationStart()函数启动 SAP2000。

3.2 初始化模型

使用 SapModel.InitializeNewModel(9)函数初始化模型，设置默认单位制为 N-mm。在 SapModel.InitializeNewModel 函数定义中可以查看单位制对应的变量值，如图 10 所示。

```
SapModel = mySapObject.SapModel
ret = SapModel.InitializeNewModel(9)
ret = SapModel.File.NewBlank()
```

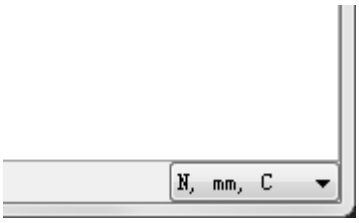

图 9 建立模型设置单位制

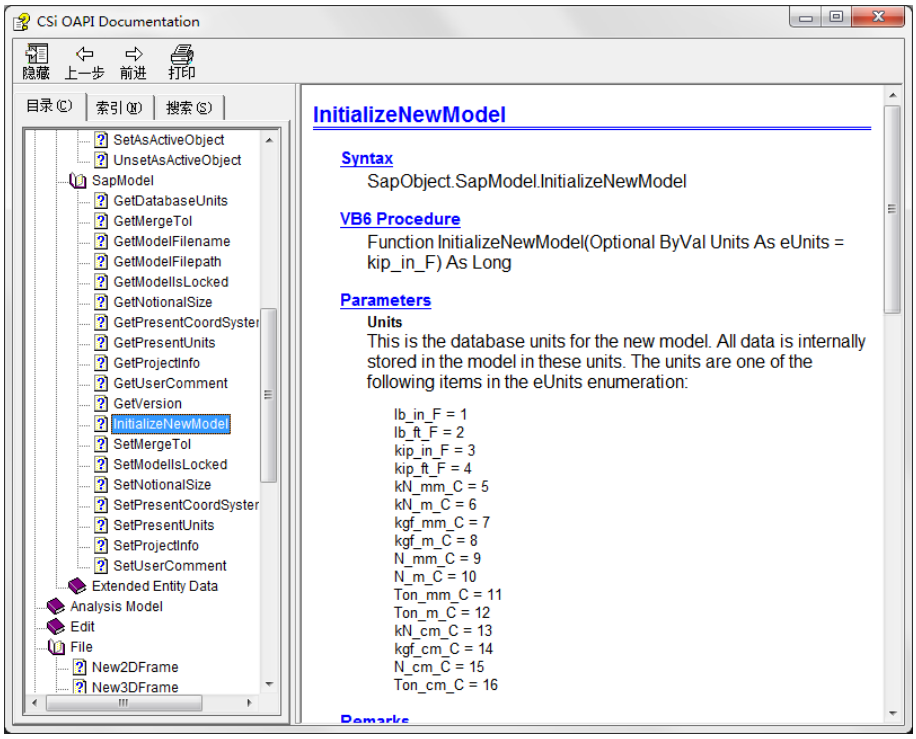

图 10 单位制对应

3.3 材料与截面定义

首先要进行材料和截面的定义。在帮助文件 CSi OAPI Functions->Definitions->Properties->Material 中查看与材料定义相关的函数，包括材料定义、参数修改、参数查询、材料删除等。材料定义有两种方式，一种是添加规范的材料，根据国家、材料类型选择，一种为自定义材料，由用户输入材料的相关参数。

使用 SapModel.PropMaterial.AddMaterial 函数添加规范材料，AddMaterial 函数中的 Name 变量为 ByRef 类型，表示该参数按引用方式传递，在函数中修改此参数的值会造成实参的值发生改变。

添加材料函数与软件操作中的材料定义函数相同，参数值不能随便设置，可以参照材料定义窗口的参数取值。参数与操作窗口的命令对应关系如图 12 所示。

```
MatName ="111"
name,ret=SapModel.PropMaterial.AddMaterial(MatName,2,"china","GB","GB50010 C40")
```

图 11 材料定义

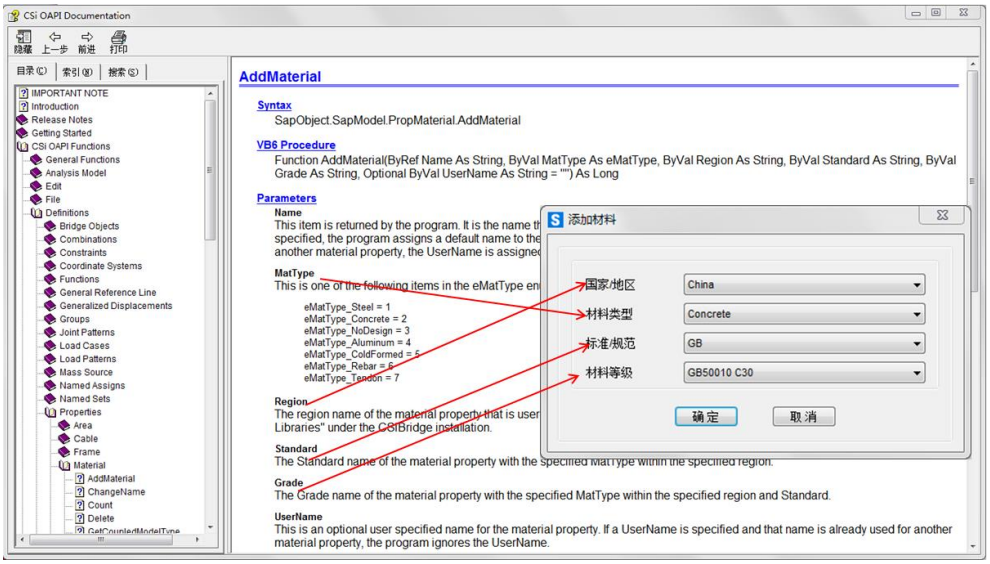

图 13 参数对应

添加自定义材料使用 SapModel.PropMaterial.SetMaterial 函数，然后根据材料类型设置参数。使用 set 开头的函数进行材料参数的设置。如 SapModel.PropMaterial.SetMPIsotropic 函数设置材料的弹性模量、泊松比、热膨胀系数，函数的具体使用可以查看函数的说明。

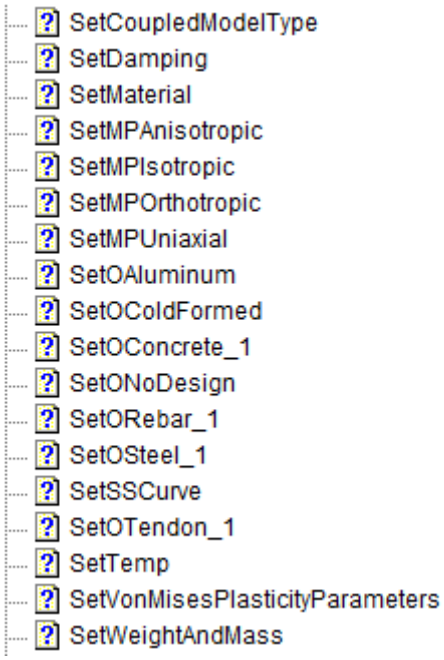

图 14 材料参数设置函数

在帮助文件 CSi OAPI Functions->Definitions->Properties->Frame 中查看与杆件截面定义相关的函数，包括截面定义、编辑、删除、参数设置等。

使用 SapModel.PropFrame.SetRectangle 函数定义矩形截面，参数包含截面名称、材料、截面高、截面宽。输入如图 14 所示代码，建立长 300mm，宽 300mm 的矩形截面。设置截面的属性修正系数，将面积放大 1000 倍。

```
ret = SapModel.PropFrame.SetRectangle('Rect1', 'C30', 300, 300)
ModValue = [1000, 0, 0, 1, 1, 1, 1, 1]
ret = SapModel.PropFrame.SetModifiers('Rect1', ModValue)
```

图 15 截面定义

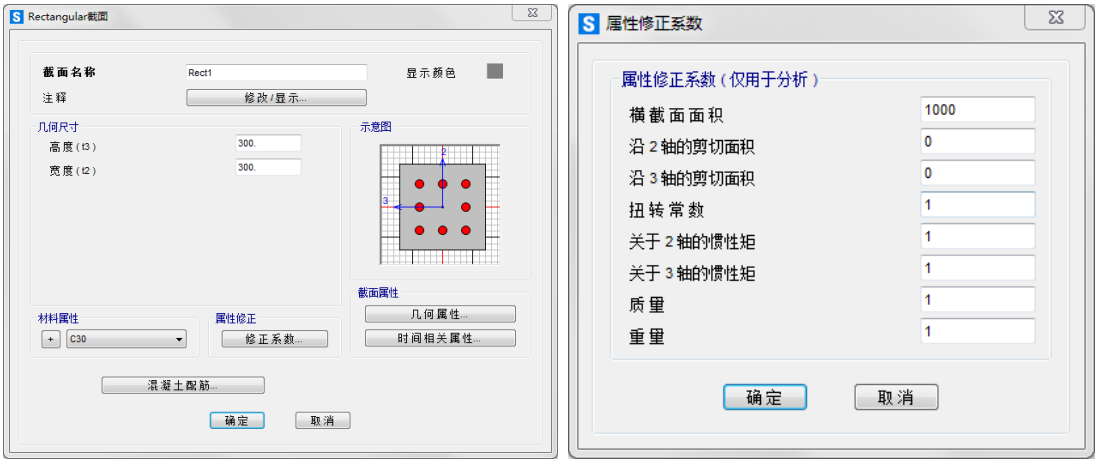

图 16 定义的截面

每种类型的截面均有单独的定义函数，具体可以查看以 set 开头的函数，如图 17 所示。SapModel.PropFrame.SetAngle 函数用于定义角钢截面，SapModel.PropFrame.SetCircle 函数用于定义圆形截面。

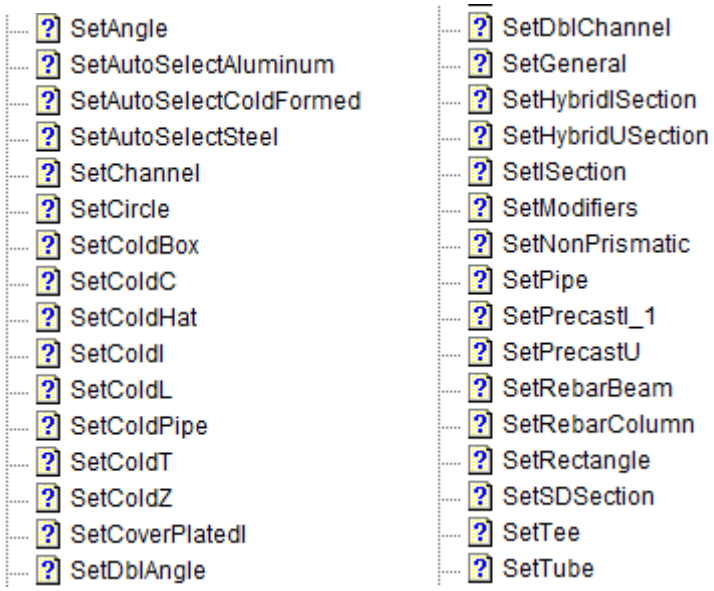

图 17 截面定义函数

3.4 构件绘制

在帮助文件 CSi OAPI Functions->Objects Model-> Frame object 中查看 Frame 单元的相关函数，包括单元绘制、荷载指定、参数调整等。

杆件绘制的方法在 API 函数中有两种，一种是通过坐标值建立杆件，一种是通过节点编号建立杆件，本文通过坐标值来添加 Frame 单元。使用函数 SapModel.FrameObj.AddByCoord 通过坐标值添加杆件。FrameName1 为 ByRef 变量，如要要得到在函数中改变后的值，需要将变量名放在等号的左侧得到改变后的值。输入如图 18 所示的代码，建立模型，如图 20 所示。

在建模过程中，可随时根据需要调整模型的单位制，使用函数 SapModel.SetPresentUnits()设置当前单位制，此后的代码输入的数据要与设置的单位相统一。

```
kN_m_C = 6
ret = SapModel.SetPresentUnits(kN_m_C)
FrameName1 = ' '
FrameName2 = ' '
FrameName3 = ' '
[FrameName1, ret] = SapModel.FrameObj.AddByCoord(0, 0, 0, 0, 0, 2.5, FrameName1, 'Rect1', '1', 'Global')
[FrameName2, ret] = SapModel.FrameObj.AddByCoord(0, 0, 2.5, 2, 0, 4, FrameName2, 'Rect1', '2', 'Global')
[FrameName3, ret] = SapModel.FrameObj.AddByCoord(-1, 0, 2.5, 0, 0, 2.5, FrameName3, 'Rect1', '3', 'Global')
```

图 18 绘制杆件

3.5 设置约束

指定构件的约束是对节点进行操作，在帮助文件 CSi OAPI Functions->Objects Model-> Point object 中查看 Point 单元的相关函数，包括节点绘制、荷载指定、约束指定等。

使用 SapModel.FrameObj.GetPoints 函数得到杆件端部的节点编号，使用 SapModel.PointObj.SetRestraint 函数设定节点的约束条件。节点的约束是一个 bool 类型的数组，设置节点在 6 个方向的自由度，true 表示约束，false 表示不约束。输入如图 19 所示的代码，为节点指定约束。生成的模型中显示节点约束，如图 20 所示。

```
PointName1 = ' '
PointName2 = ' '
Restraint = [True, True, True, True, False, False]
[PointName1, PointName2, ret] = SapModel.FrameObj.GetPoints(FrameName1, PointName1, PointName2)
ret = SapModel.PointObj.SetRestraint(PointName1, Restraint)

Restraint = [True, True, False, False, False, False]
[PointName1, PointName2, ret] = SapModel.FrameObj.GetPoints(FrameName2, PointName1, PointName2)
ret = SapModel.PointObj.SetRestraint(PointName2, Restraint)

ret = SapModel.View.RefreshView(0, False)
```

图 19 设置约束条件

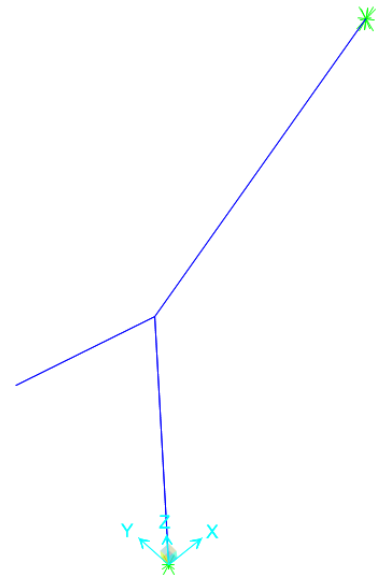

图 20 SAP2000 模型

3.6 荷载定义与指定

定义荷载模式，并对构件指定荷载。在帮助文件的 CSi OAPI Functions->Definitions->Load patern 部分查看荷载模式定义相关的函数，包括荷载模式的定义、编辑、删除等。使用 SapModel.LoadPatterns.Add 函数添加荷载模式定义。定义荷载模式后，

自动生成同名的荷载工况。

根据节点编号指定节点荷载，使用函数 SapModel.PointObj.SetLoadForce 为节点指定集中荷载为-2kN。根据杆件单元编号，指定均布荷载，使用函数 SapModel.FrameObj.SetLoadDistributed 函数指定杆件上的均布荷载。输入如图 21 所示的代码，增加两个活荷载工况，工况名称为 Live、Live2，为节点和构件指定荷载，模型显示如图 22 所示。

```
LTYPE_LIVE = 3
ret = SapModel.LoadPatterns.Add('Live', LTYPE_LIVE, 0, True)
ret = SapModel.LoadPatterns.Add('Live2', LTYPE_LIVE, 0, True)

[PointName1, PointName2, ret] = SapModel.FrameObj.GetPoints(FrameName3, PointName1, PointName2)
PointLoadValue = [0, 0, -2, 0, 0, 0]
ret = SapModel.PointObj.SetLoadForce(PointName1, 'Live', PointLoadValue)
ret = SapModel.FrameObj.SetLoadDistributed(FrameName3, 'Live', 1, 10, 0, 1, 1.5, 1.5)


[PointName1, PointName2, ret] = SapModel.FrameObj.GetPoints(FrameName3, PointName1, PointName2)
PointLoadValue = [0, 0, -4, 0, -17, 0]
ret = SapModel.PointObj.SetLoadForce(PointName2, 'Live2', PointLoadValue)
ret = SapModel.FrameObj.SetLoadDistributed(FrameName2, 'Live2', 1, 11, 0, 1, 2, 2)
```

图 21 指定荷载

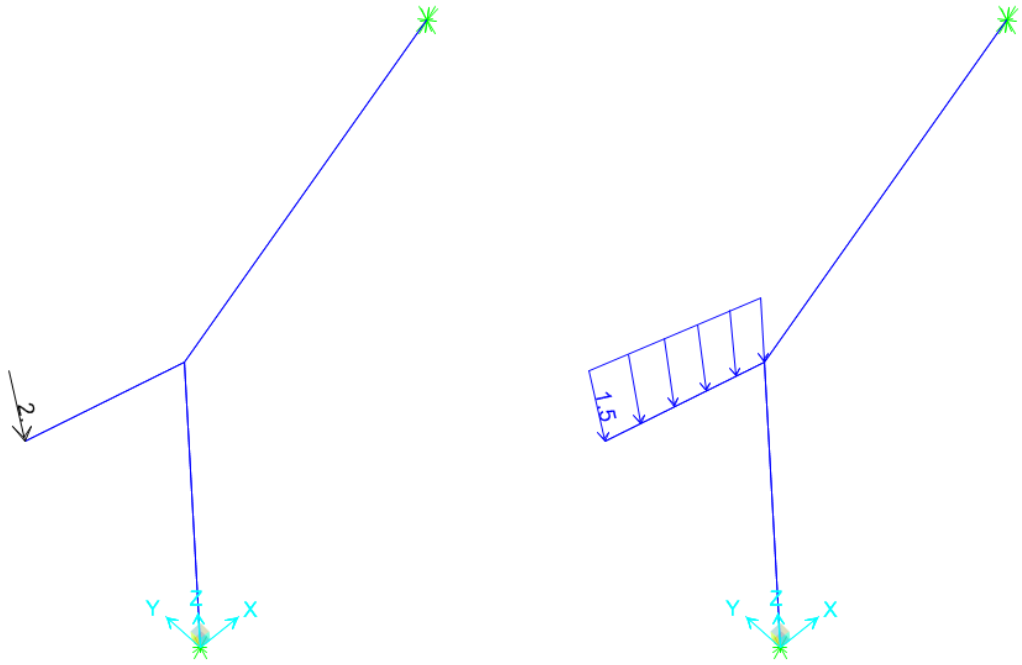

图 22 荷载显示

3.7 运行分析返回结果

保存建立的模型，并进行分析，在分析完成后提取工况下的位移结果。使用 SapModel.Results.JointDispl 函数得到节点位移结果，在输出位移结果前，要先用 SapModel.Results.Setup.DeselectAllCasesAndCombosForOutput 函数，清除设置的荷载工况，使用 SapModel.Results.Setup.SetCaseSelectedForOutput 函数设置要输出结果的荷载工况，再使用节点位移函数输出结果。输入如图 23 所示代码，保存建立的模型，并运行分析，分析完成后读取某一节点所有工况的位移，并显示。

节点位移打印结果如图 24 所示，与 SAP2000 表格中输出的结果一致。

```
ret = SapModel.File.Save('d:\\frameAPI.sdb')
ret = SapModel.Analyze.RunAnalysis()

SapResult = [0, 0, 0, 0, 0, 0, 0]
[PointName1, PointName2, ret] = SapModel.FrameObj.GetPoints(FrameName2, PointName1, PointName2)
LoadCaseS = [];NumberNames = 0
[NumberNames, LoadCaseS, ret] =SapModel.LoadCases.GetNameList_1(NumberNames,LoadCaseS,1)

for i in range(0,NumberNames):
    NumberResults = 0;    Obj = [];    Elm = []
    ACase = [];    StepType = [];    StepNum = []
    U1 = [];    U2 = [];    U3 = []
    R1 = [];    R2 = [];    R3 = []
    ObjectElm = 0
    ret = SapModel.Results.Setup.DeselectAllCasesAndCombosForOutput()
    ret = SapModel.Results.Setup.SetCaseSelectedForOutput(LoadCaseS[i])
    [NumberResults, Obj, Elm, ACase, StepType, StepNum, U1, U2, U3, R1, R2, R3, ret] = SapModel.Results.JointDispl(
        PointName2, ObjectElm, NumberResults, Obj, Elm, ACase, StepType, StepNum, U1, U2, U3, R1, R2, R3)
    print("工况:",LoadCaseS[i],"节点:",PointName2,"位移: ",U1, U2, U3, R1, R2, R3)
```

图 23 运行分析提取结果

```
工况: DEAD 节点: 3 位移:  (0.0,) (0.0,) (-0.0002748965509238543,) (0.0,) (0.00016059075694299427,) (0.0,)
工况: Live 节点: 3 位移:  (0.0,) (0.0,) (3.53624948557006e-05,) (0.0,) (3.5364207820795507e-06,) (0.0,)
工况: Live2 节点: 3 位移:  (0.0,) (0.0,) (1.2853971193319009e-05,) (0.0,) (0.00013503060493833926,) (0.0,)
```

图 24 结果打印

	Joint Text	OutputCase	CaseType Text	U1 m	U2 m	U3 m	R1 Radians	R2 Radians	R3 Radians
▶	3	DEAD	LinStatic	0	0	-0.000275	0	0.000161	0
	3	Live	LinStatic	0	0	3.5E-05	0	3.536E-06	0
	3	Live2	LinStatic	0	0	1.3E-05	0	0.000135	0

图 25 SAP2000 表格结果

4 与 VBA 对比

VBA 不需要单独的安装，相对于专业的开发软件，简单易学，了解基本的语法后，就可以编写代码，如参数化建模，结果处理等。VBA 可以在不同的软件中集成，如 Word、Excel 等。当用于结果处理时，在 Excel 中进行开发，将数据提取到 Excel 表格，使用 Excel 函数对数据进行处理，输出图表等，便于结果的提取与处理。

Python 支持调用各种库文件，实现 COM 组件的调用、结果处理、图形显示等。Python 语言简单易学，通过加载库文件，实现各种类型的计算，对于建模、结果读取等方便便捷。但 Python 编写的代码运行速度较低，如果有大量的数据计算，可以将核心的计算部分使用 C++语言编写来提高运行效率。

5 结束语

Python 代码易于编写与理解，新手也容易上手。在了解 Python 语言特点和熟悉 SAP2000 的操作后，就可以通过 API 编写实用的工具，将日常工作中繁琐的处理工作简单化，提高工作效率。