ChemAppPy

Python Interfaces to ChemApp

(the best of two worlds)

Johan Zietsman, Christoff Kok

1 July 2016

GTT Users Meeting 2016

Agenda

- Background
- ChemApp - Pros and Cons
- Python?
- **`ChemAppPy`** Overview
- **`ChemAppPy raw`** Interface
- **`ChemAppPy friendly`** Interface
- **`ChemAppPy`** Documentation
- Python … just another language?
- Conclusion

Background

- Two affiliations
  - Ex Mente
  - University of Pretoria
- Thermochemistry an essential tool in pyrometallurgy
- FactSage
  - Makes thermochemistry come to life 👍
  - Doing many calculations can be very time consuming 👎
  - Limited flexibility (determined by UI) 👎
- ChemApp
  - Virtually unlimited flexibility 👍
  - Requires C, Fortran (compiled languages) 👎
  - May be difficult to learn 👎
  - May be slow to develop algorithms 👎

Background

- We needed:
  - Complete flexibility
  - Rapid development solution
  - A rich, productive environment
  - Must be possible to master the interface quickly

… **`ChemAppPy`** was born

ChemApp – Pros and Cons

- API for thermochemical calculations
- Very powerful and useful
- Total of 75 functions

ChemApp – Pros and Cons

```
tqbond   tqce   tqcel   tqcen   tqcenl   tqcio
tqclim  tqclos  tqcprt  tqcsc  tqcsp  tqcspc
tqcsu  tqerr  tqgdpc  tqgetr  tqgio  tqgnlc
tqgnp  tqgnpc  tqgnsc  tqgsp  tqgspc  tqgsu
tqgted   tqgthi   tqgtid   tqgtlc   tqgtnm
tqgtpi  tqgtrh  tqini  tqinlc  tqinp  tqinpc
tqinsc   tqlite   tqmap   tqmapl   tqmodl
tqnolc   tqnop   tqnopc   tqnosc   tqnosl
tqopen   tqopna   tqopnb   tqopnt   tqpcis
tqrbin   tqrcst   tqremc   tqrfil   tqsetc
tqshow   tqsize   tqstca   tqstec   tqstpc
tqstrm   tqstsc   tqsttp   tqstxp   tqused
tqvers  tqwstr
```

ChemApp – Pros and Cons

- API for thermochemical calculations
- Very powerful and useful
- Total of 75 functions
- Functions are very compact
  - `tqsetc` can do 21 different things
- API can
  - Difficult to learn
  - Difficult to remember
- Requires C or Fortran
  - Not so many engineers in SA know this
  - Compilation is a time-consuming step
- Multiple lines of code per task

ChemApp – Pros and Cons

Open a thermochemical data file (9 lines):

```
LI noerr;

tqini(&noerr);
tqopna("cosi.dat",10,&noerr);
if (noerr) {
      printf("ERROR: Cannot open data-file.");
      exit(noerr);
}
tqrfil(&noerr);
tqclos(10,&noerr);
```

ChemAppPy raw

Open a thermochemical data file (4 lines):

```
tqini()
tqopna("cosi.dat",10)
tqrfil(noerr)
tqclos(10)
```

ChemAppPy friendly

Open a thermochemical data file (1 lines):

```
ThermochemicalSystem.load_data("cosi.dat")
```

Python?

`ChemAppPy` is developed for Python

Python

- Open-source programming language
- Started in 1980s
- Freely available
- Taught in 2nd year engineering at UP
- Large community of users and developers
- Well-suited for science and engineering

Why use ChemApp in Python?

- Accessibility (open source)
- Cross-platform (Windows, Mac, Linux)
- Simplicity (language)
- Rapid development
- Power (e.g. object orientation)
- Versatility (the perfect “glue” language)
- Reliability (readable, testable)
- Support (large community, e.g. stackoverflow)
- **Many** packages available (math, numerics, etc.)

ChemAppPy Overview

- What's in the box?
- Raw interface

  'Direct' translation of ChemApp API to Python
- Friendly interface

  More verbose and clear interface
- Automated errors (exceptions)
- On-line documentation
- Jupyter Notebook Examples

ChemAppPy raw Interface

- Example

ChemAppPy friendly Interface

- More verbose
- More clear to understand (hopefully)
- Easier to learn and remember
- Divided into categories of functions:
  - Info
  - ThermochemicalSystem
  - Units
  - EquilibriumCalculation
  - StreamCalculation
  - PhaseMapCalculation

ChemAppPy friendly Interface

Abbreviations:

```
eq  : equilibrium
ph  : phase
phs : phases
pc  : phase constituent
pcs : phase constituents
sc  : system component
scs : system components
```

ChemAppPy friendly Interface

Abbreviations:

A : amount
CP : heat capacity
E : energy
G : Gibbs energy
H : enthalpy
IA : incoming amount
MU : chemical potential
P : pressure
S : entropy
T : temperature
V : volume
VT : total volume
X : fraction
XT : mole fraction

ChemAppPy friendly Interface

- Example

`ChemAppPy friendly` Interface

- ChemApp in Fortran:
  - 98 lines of code
- **`ChemAppPy raw`**:
  - 63 lines of code
  - 35% less than Fortran
- **`ChemAppPy friendly`**:
  - 42 lines of code
  - 57% less than Fortran
  - 33% less than **`ChemAppPy raw`**

ChemAppPy Demonstration

- Documentation
- Interpreted language
- Plotting

Python … just another language?

- Perhaps not
- Rich environment for science and engineering
- SciPy (integration, optimisation interpolation, fourier, stats, etc)
- Numpy (linear algebra)
- Matplotlib (plotting)
- Pandas (analysis on large data sets)
- Jupyter (interactive computing)
- Much more …
- Great synergy when adding ChemApp

Conclusion

- Having ChemApp available in Python
  - Makes it more accessible
  - Enhances productivity
  - Reduces the learning curve for ChemApp
- We have been focusing on:
  - Pre-processing stage
  - Solving stage
- Next will be:
  - Post-processing routines

ChemAppPy

“I’ve been using the friendly version of ChemApp for Python for a couple of weeks now and it has really changed my life…

It has taken out the complexity of writing complicated (for a non C-person) C-code …

made pre- and post-processing so much easier, since I’m using Python for many things already. …

I’ve been finishing projects much more rapidly than in the past…”

Markus Erwee, Principal Engineer, Mintek, South Africa