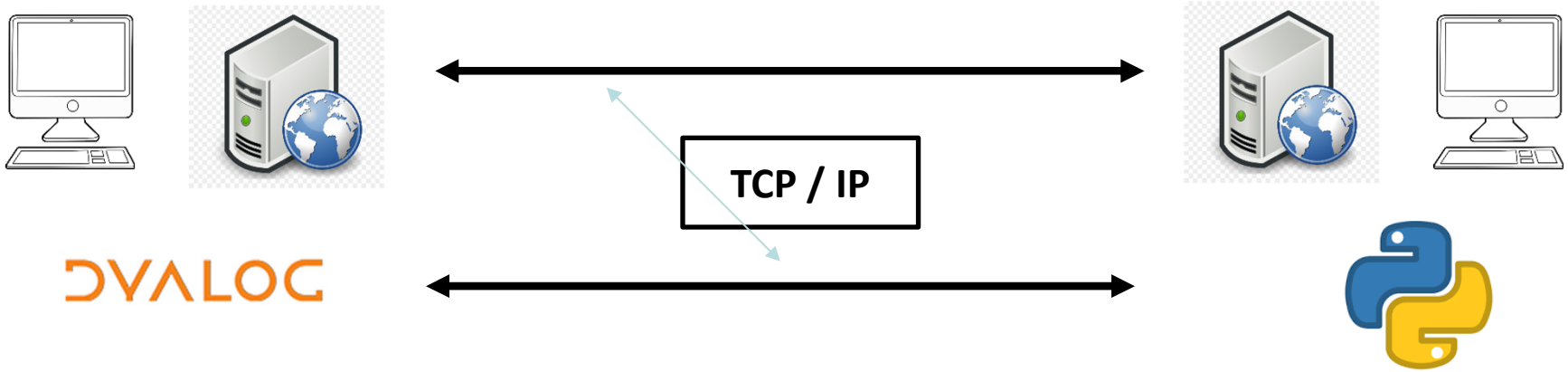
Dittrich & Partner Consulting GmbH

APL trifft auf Python

Technische Anbindung von Python an APL



- Dyalog APL als Client, Python als Server und umgekehrt
- Master- / Slave – Architektur
- Kommunikation über TCP (CONGA (Namespace IPC), IPC.py)
- pynapl enthält APL- und Python-Code
- Nutzung von Python-Funktionalitäten in Dyalog APL und umgekehrt

Technische Anforderungen für Anbindung Python

- Dyalog APL 16.0 oder höher (Unicode Edition 64 bit)
- Python 2.7.9 oder höher bzw. 3.4 oder höher
- pynapl – Freies Paket auf Github für Verbindung von Dyalog APL und Python
- CONGA – Tool in Dyalog APL für Remote-Verbindungen
- SALT – APL-Code in binärer Datei

Anbindung Python an APL

- Einsatz von SALT (Simple APL Library Toolkit)
  - Quellcode in Unicode-Textdatei Py.dyalog
  - Python-Skripte müssen im gleichen Ordner liegen wie Salt-Datei
- Namespace Py
  - Klassen und Methoden zur Ausführung von Python in APL
  - Start von Python im Hintergrund über neue Instanz der Klasse Py.Py
- Namespace IPC
  - Aufbau der TCP-Kommunikation
  - Einsatz von Conga
- Python-Skripte
  - APLBridgeSlave.py
  - etc.

Aufrufe von Python aus APL - Verbindungsaufbau

- Start der APL-Schnittstelle in SALT-Datei Py.dyalog

```
]load Py
#.Py
```

- Start von Python im Hintergrund und Aufbau der TCP-Verbindung durch Erstellen einer neuen Instanz der Klasse Py.Py

```
py←⍞NEW Py.Py
py
#.[Py]

pypath
C:\ProgramData\anaconda3\python.exe

arg
"C:\Daten\Dyalog\Python\pynapl-master\pynapl\APLBridgeSlave.py" TCP 65535
```

Aufrufe von Python aus APL - Ausführung

- Ausführen von Python-Statements durch Aufruf der Exec-Funktion

```
py.Exec 'import antigravity'
```

- Ausführen von Python-Ausdrücken durch Aufruf der Eval-Funktion

```
'val' py.Set ⍳5
py.Eval 'val'
1 2 3 4 5
```

Python-Funktionen und Module in APL nutzen

- Definition von APL-Funktionen, die Python-Funktionen aufrufen mit Hilfe der Funktionen Call und CallVec aus dem Namespace PyFn

```
py.Exec 'import webbrowser'
sp←(py.PyFn 'webbrowser.open').Call

sp 'http://www.dpc.de'

sp 'http://www.apl-germany.de'
```

- Python- Objekte und Module in APL nutzen

```
py.Exec 'import os'
os←py.Eval 'os‘

+os.getlogin ⍳0
Christian Bäumer

+os.getpid ⍳0
13048
```

```
os←py.Import 'os‘

+os.getlogin ⍳0
Christian Bäumer

+os.getpid ⍳0
13048
```

Anbindung APL an Python

- Zugriff auf Python-Module zum Verbindungsaufbau
  - APL.py
  - APLPyConnect.py
  - RunDyalog.py
- Zugriff auf Salt-Dateien, die im gleichen Verzeichnis erwartet werden
  - Py.dyalog
  - IPC.dyalog
- Weitere Python-Module
  - Array.py
  - PyEvaluator.py
  - ObjectWrapper.py
  - etc.

Aufrufe von APL aus Python - Verbindungsaufbau

- Initialisierung der Verbindung von Python zu APL durch Aufruf der Funktion APL aus dem Modul APL.py

```
from pynapl import APL
apl = APL.APL()
```

- Ausführung von APL-Code aus Python heraus mit der Funktion Eval

```
>>> apl.eval("2+2")
4
>>> apl.eval("⎕A")
u'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

>>> apl.eval("'2+2' py.Eval 0") # round trip
4
```

APL-Funktionen in Python nutzen (1)

- Die Funktion fn ermöglicht den Import von APL-Funktionen nach Python

- Die APL-Funktionen können dabei niladisch, monadisch oder dyadisch sein

```
>>> aplsum = apl.fn("+/")
>>> aplsum([1,2,3,4,5])
15
>>> aplsum(3, [1,2,3,4,5])
[6, 9, 12]
```

- Einbindung einer mit dfns definierten APL-Funktion in Python

```
>>> factorial = apl.fn("""
{ ω≤0:1
  ω×∇ω-1
}
""")
>>> factorial(5)
120
```

APL-Funktionen in Python nutzen (2)

- Einbindung einer klassisch definierten APL-Funktion in Python

```
>>> foo = apl.tradfn("""
r←foo x
r←x+x
""")
>>> foo(5)
10
>>> apl.fn("foo")(5)
10
```

APL-Operatoren in Python nutzen

- Einbindung einer klassisch definierten APL-Funktion in Python

```
>>> scan = apl.op("\\")

>>> apl_add = apl.fn("+")

>>> apl_sumscan = scan(apl_add)


>>> apl_sumscan([1,2,3])
[1, 3, 6]
```

DPC GmbH – Ihr Ansprechpartner

Christian Bäumer
Geschäftsführer
Aktuar DAV

Dittrich & Partner Consulting GmbH
Prinzenstraße 2a
42697 Solingen

Telefon: +49 (212) 26 06 6-22
Mobil: +49 170 86 14 781
christian.baeumer@dpc.de

Tel. +49 (212) 26 06 6-0
E-Mail: info@dpc.de
www.dpc.de