A ReSTful Interface for CubeSats

Shaun Houlihan

Pumpkin, Inc.

CubeSat Software Interfaces

- Many mission unique protocols
- Low level manipulation
- Protocol libraries
- Platform lock
- Interface documents
- Etc...

```
#include <stdio.h>
#define m 21
#define o(l, k) for(l=0; l<k; l++)
#define n(k) o(T, k)


              int E,L,O,R,G[42][m],h[2][42][m],g[3][8],c
              [42][42][2],f[42]; char d[42]; void v( int
              b,int a,int j){ printf("\33[%d;%df\33[4%d"
              "m  ",a,b,j); } void u(){ int T,e; n(42)o(
              e,m)if(h[0][T][e]-h[1][T][e]){ v(e+4+e,T+2
              ,h[0][T][e]+1?h[0][T][e]:0); h[1][T][e]=h[
              0][T][e]; } fflush(stdout); } void q(int l
                            ,int k,int p){
                            int T,e,a; L=0
                            ; O=1; while(O
                            ){ n(4&&L){ e=
                            k+c[l] [T][0];
                            h[0][L-1+c[l][
                            T][1]][p?20-e:
e]=-1; } n(4){                                          e=k+c[l][T][0]; a=L+c[l][T][
1]+1; if(a==42                                          || h[0][a][p?20-e:e]+1){ O=0
; } } n(4){ e=                                          k+c[l][T][0]; h[0][L + c[l][
T][1]][p?20-e:                                          e]=g[1][f[p?19+l:l]]; } L++;
u(); } n(42) {                                          o(e,m)if(h[0][T][e]<0)break;
o(a, m&&e==m){                                          for(L=T; L; L--) { h[0][L][a
]=h[0][L-1] [a                                          ]; } h[0][0][a]=-1; } } u();
}int main(){ int T,e,t,r,i,s              ,D,V,K; printf("\33[2J\33[?25l"); n(8)g[i=
1][T]=7-T; R--; n(42) o(e,m)              G[T][e]--; while(fgets(d,42,stdin)) { r=++
R; n(17){ e=d[T]-48; d[T]=0;              if ((e&7)==e) { g[0][e] ++; G[R][T+2]=e; }
} } n(8)if(g[0][7-T]){ t=g[i              ][O]; g[i][O++]=g[i][T]; g[i][T]=t; } n(8)
g[2][g[i][T]]=T; n(R+i)o(e,m              )if(G[T][e]+i) G[T][e]=g[2][G[T][e]]; n(19
)o(t,2){ f[T+t+T]=(T["+%#,4"              "5>GP9$5-,#C?NX"]-35)>>t*3&7; o(e,4){ c[T]
[e][t]=("5'<$=$8)Ih$=h9i8'9"              "t=)83)l4(99(g9>##>4(" [T+t+T]-35)>>e*2&3;
} } n(15) { s=T>9?m:(T&3)-3?15:36;o(e,s)o(t,2)c[T+19][e][t]="6*6,8*6.608.6264826668\
865::(+:0(6+6-6/8.61638065678469.:88))()3(6.8*6.608.6264826668865:+:4)-*6-6/616365.\
```

Software Interfaces Ideally..

- Universal
- Language independent
- Platform independent
- High Level
- Able to leverage open source code
- Simple to document

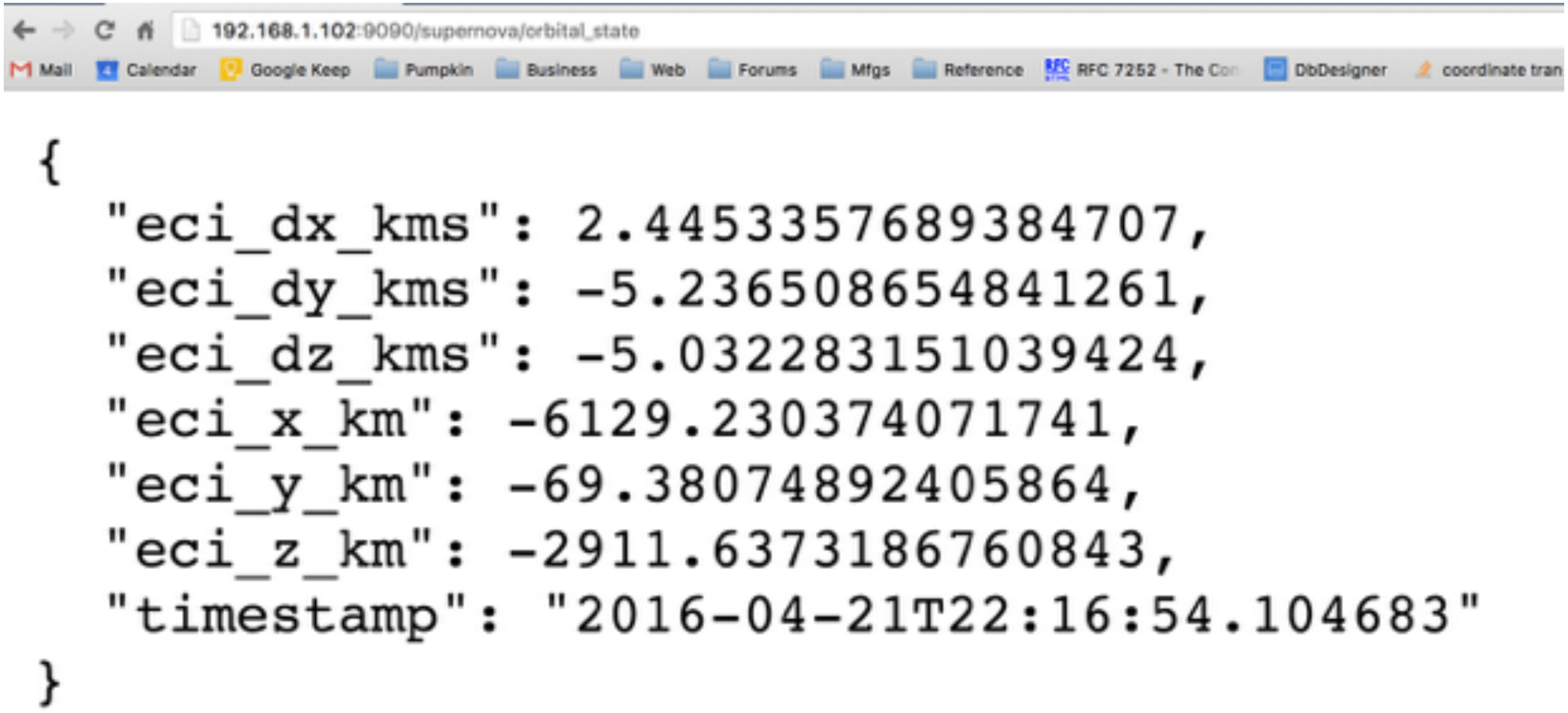
- Where would we find something like that?

ReST - “The software architectural style of the Web”[1]

- **Representational State Transfer**
- **ReSTful system constraints:**
  1. Client-Server
  2. Stateless
  3. Cache
  4. Uniform Interface
  5. Layered
  6. Code on Demand
- **Protocol + Data Format not explicitly stated**
  - But ***HTTP*** **+** ***JSON*** are web standard

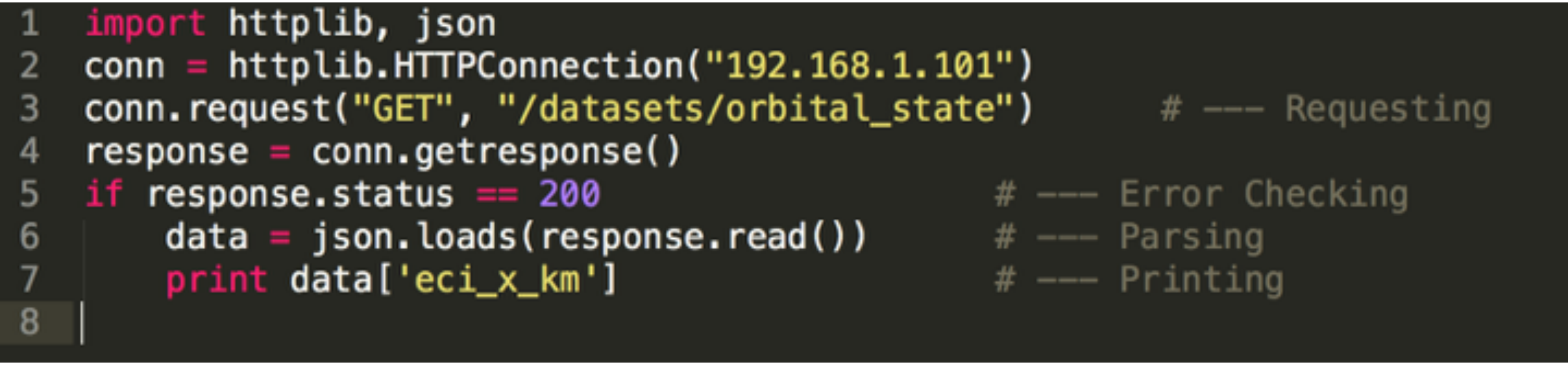
What does using ReST look like?

- Browser demo



```
{
  "eci_dx_kms": 2.4453357689384707,
  "eci_dy_kms": -5.2365086548412610,
  "eci_dz_kms": -5.0322831510394240,
  "eci_x_km": -6129.23037407174100,
  "eci_y_km": -69.380748924058640,
  "eci_z_km": -2911.63731867608430,
  "timestamp": "2016-04-21T22:16:54.104683"
}
```

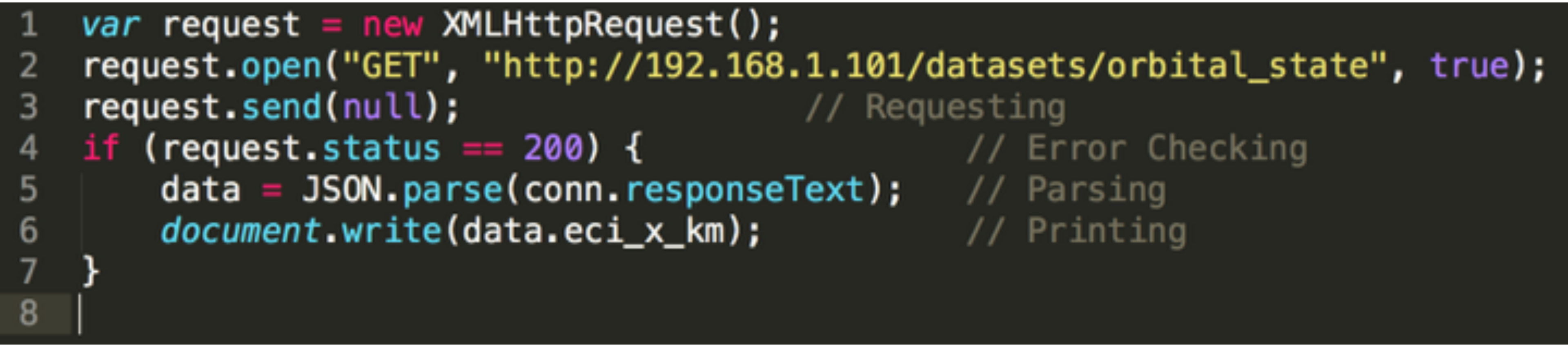
Payload Example - Python



```python
import httplib, json
conn = httplib.HTTPConnection("192.168.1.101")
conn.request("GET", "/datasets/orbital_state")      # --- Requesting
response = conn.getresponse()
if response.status == 200                   # --- Error Checking
    data = json.loads(response.read())      # --- Parsing
    print data['eci_x_km']                  # --- Printing

```

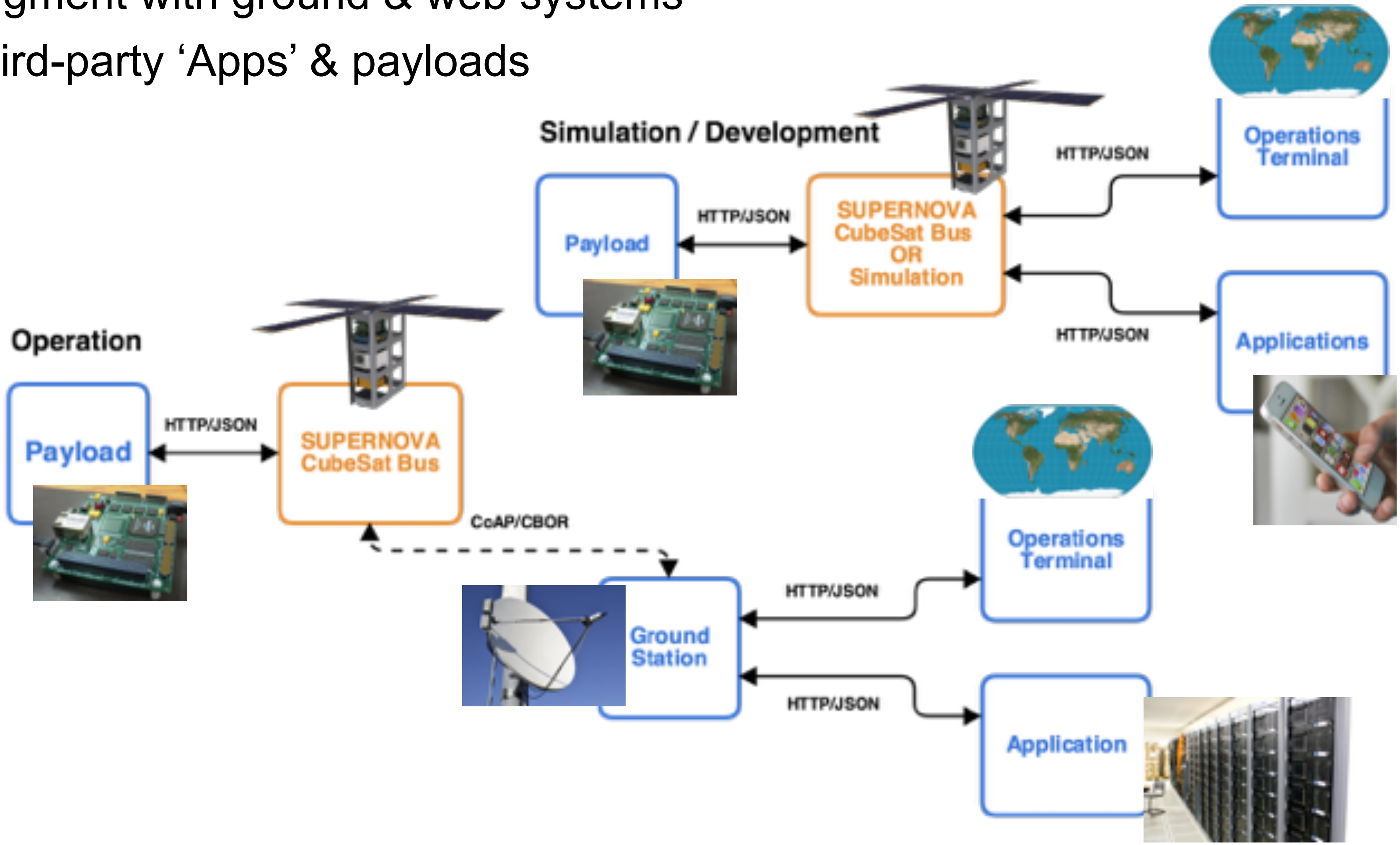
Payload Example - JavaScript



```javascript
var request = new XMLHttpRequest();
request.open("GET", "http://192.168.1.101/datasets/orbital_state", true);
request.send(null);                 // Requesting
if (request.status == 200) {                // Error Checking
    data = JSON.parse(conn.responseText);   // Parsing
    document.write(data.eci_x_km);          // Printing
}
```

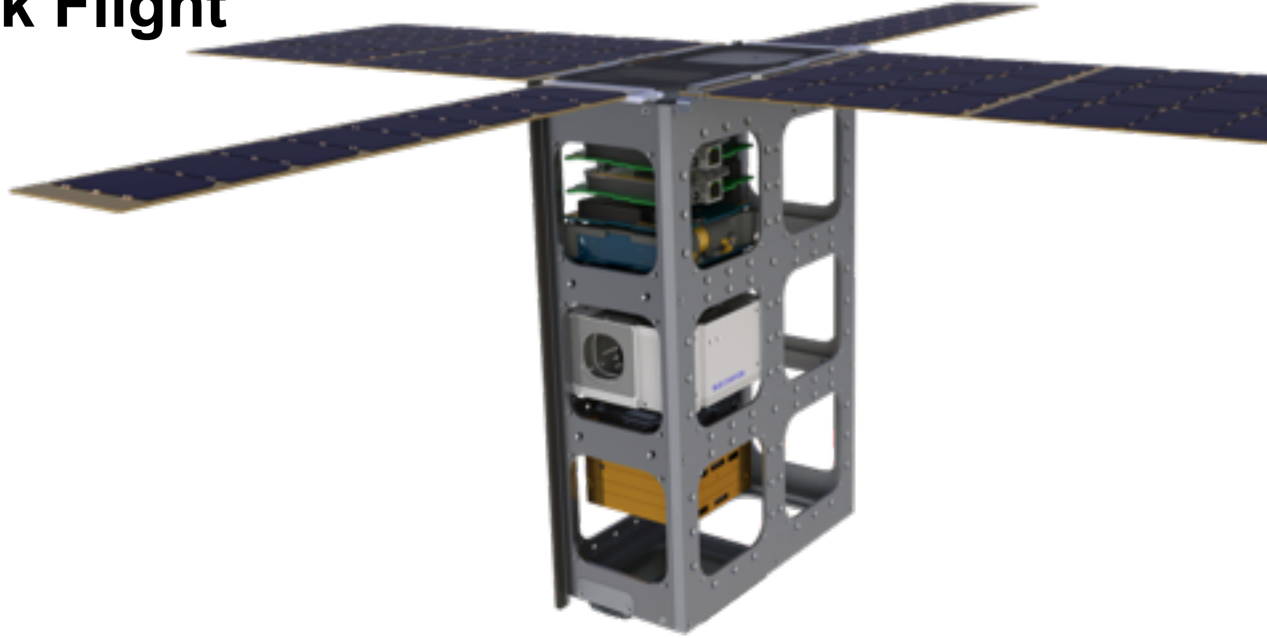
ReST/HTTP Extensibility

- Augment with ground & web systems
- Third-party 'Apps' & payloads



ReST Implementation - Hardware

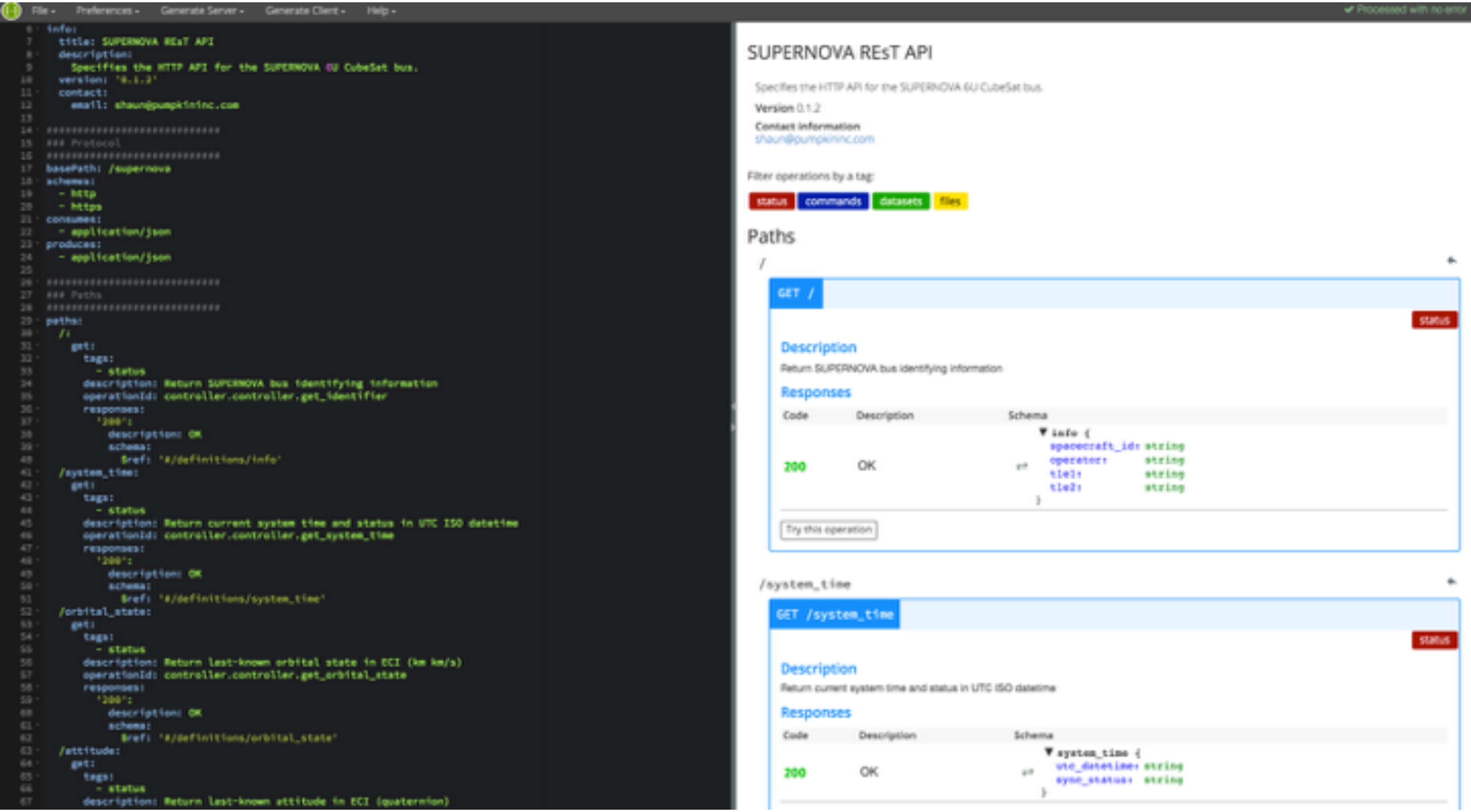
- **SUPERNOVA Bus**
  - 64W Solar Power
  - Ethernet Switch (in development)
  - Up to 4U Payload Volume
- **BeagleboneBlack Flight Computer**
  - ~2W
  - 1GHz ARM CPU
  - Linux



ReST Implementation - Software

- **Augments high-TRL bus flight software**
  - Low-level interface still available
- **“API First”**
  - Describe interface with using “Swagger.io” standard & tools
    - http://www.swagger.io
  - Auto-generate documentation from spec
  - Auto-generate HTTP server stubs from spec
  - Fill in the callbacks

Swagger Document Generation



```
info:
  title: SUPERNOVA REsT API
  description:
    Specifies the HTTP API for the SUPERNOVA 6U CubeSat bus.
  version: '0.1.2'
  contact:
    email: shaun@pumpkininc.com

###########################
### Protocol
###########################
basePath: /supernova
schemes:
  - http
  - https
consumes:
  - application/json
produces:
  - application/json

###########################
### Paths
###########################
paths:
  /:
    get:
      tags:
        - status
      description: Return SUPERNOVA bus identifying information
      operationId: controller.controller.get_identifier
      responses:
        '200':
          description: OK
          schema:
            $ref: '#/definitions/info'
  /system_time:
    get:
      tags:
        - status
      description: Return current system time and status in UTC ISO datetime
      operationId: controller.controller.get_system_time
      responses:
        '200':
          description: OK
          schema:
            $ref: '#/definitions/system_time'
  /orbital_state:
    get:
      tags:
        - status
      description: Return last-known orbital state in ECI (km km/s)
      operationId: controller.controller.get_orbital_state
      responses:
        '200':
          description: OK
          schema:
            $ref: '#/definitions/orbital_state'
  /attitude:
    get:
      tags:
        - status
      description: Return last-known attitude in ECI (quaternion)
```

SUPERNOVA REsT API

Specifies the HTTP API for the SUPERNOVA 6U CubeSat bus.

Version 0.1.2

Contact information
shaun@pumpkininc.com

Filter operations by a tag:

Paths

/

Description

Return SUPERNOVA bus identifying information

Responses

Code	Description	Schema

200	OK	▼ info {
		spacecraft_id: string
		operator: string
		tle1: string
		tle2: string
		}

/system_time

Description

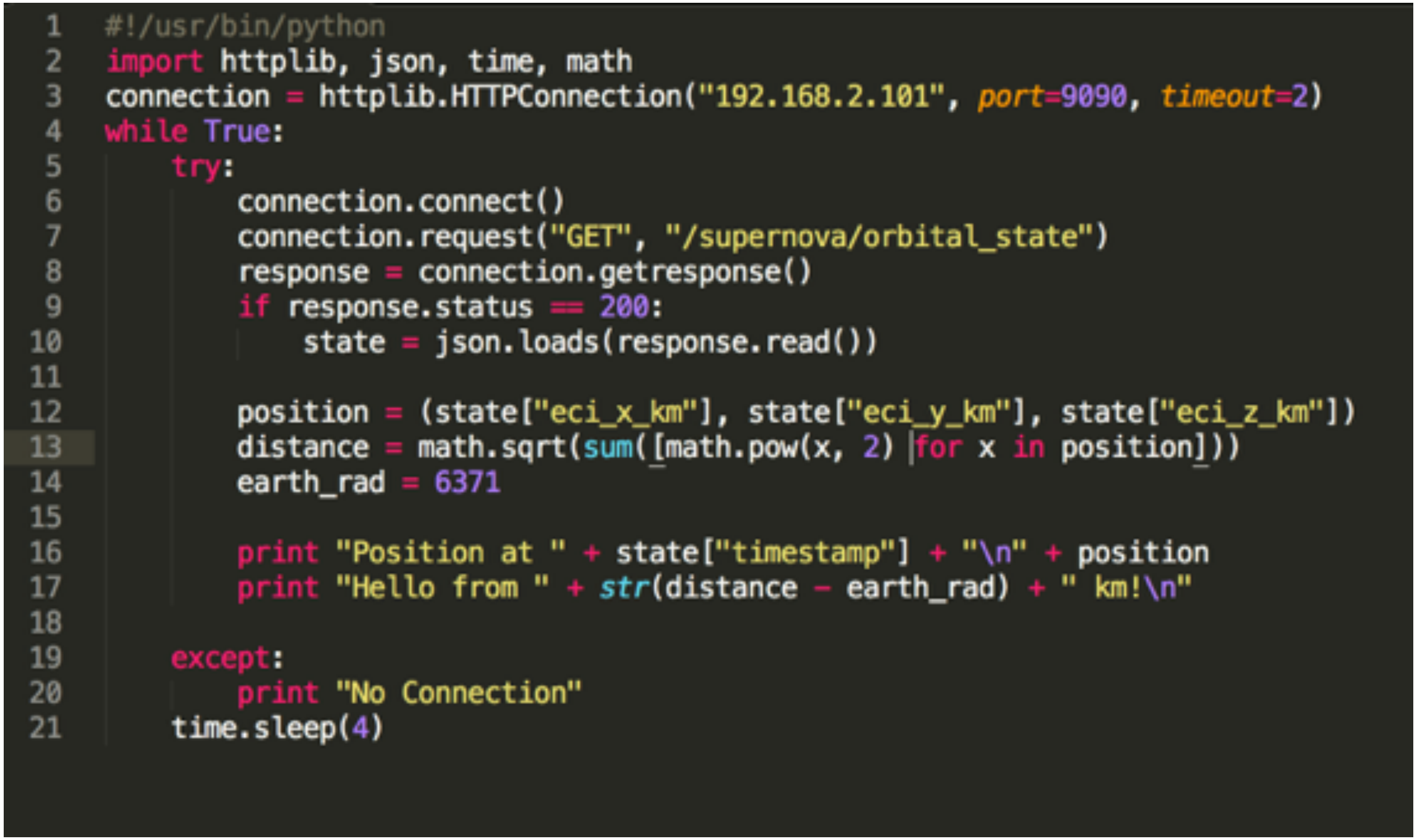
Return current system time and status in UTC ISO datetime

Responses

Code	Description	Schema

200	OK	▼ system_time {
		utc_datetime: string
		sync_status: string
		}

Payload Demo...



```
#!/usr/bin/python
import httplib, json, time, math
connection = httplib.HTTPConnection("192.168.2.101", port=9090, timeout=2)
while True:
    try:
        connection.connect()
        connection.request("GET", "/supernova/orbital_state")
        response = connection.getresponse()
        if response.status == 200:
            state = json.loads(response.read())

        position = (state["eci_x_km"], state["eci_y_km"], state["eci_z_km"])
        distance = math.sqrt(sum([math.pow(x, 2) for x in position]))
        earth_rad = 6371

        print "Position at " + state["timestamp"] + "\n" + position
        print "Hello from " + str(distance - earth_rad) + " km!\n"

    except:
        print "No Connection"
    time.sleep(4)
```

Conclusions + Future

- **ReST fits Pumpkin approach to Nanosats**
  - Open, Standardized, Accessible, Scalable, Modular, Customizable
  - “Fly Your Laptop”
- **Development at Pumpkin:**
  - HTTP payload interface to SUPERNOVA
  - SUPERNOVA Bus Simulator

PUMPKIN™

SPACE SYSTEMS

Q&A Session

Thank you for attending this Pumpkin presentation at the 2016 CubeSat Developer’s Workshop!

Appendix

- **Further Reading**
  - https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
  - http://whatisrest.com
  - http://coap.technology
  - http://cbor.io
  - http://swagger.io
- **Speaker information**
  - Shaun Houlihan is an Engineer at Pumpkin involved in developing SUPERNOVA hardware and software. Before joining Pumpkin he worked in the aerospace and consumer electronics industries. Contact Shaun at shaun@pumpkininc.com.
- **CubeSat Kit information**
  - More information on Pumpkin's CubeSat Kit can be found at http://www.cubesatkit.com/. Patented and Patents pending.
- **Copyright**

Notice

This presentation is available online at:

www.pumpkininc.com/content/doc/press/20160420_Pumpkin_CSDWSLO_2016.pdf
First presented at the CubeSat Developer’s Workshop in San Luis Obispo, California on Friday, April 22, 2016