Streaming Web-Services for Calculating Live Hydrological Derivatives

Master Thesis Defense

Christian Autermann

Supervisors: Edzer Pebesma (IfGI), Jordan Read (USGS CIDA)

January 31, 2014

Outline

1. Lake Analyzer
2. Matlab WPS
3. Lake Analyzer WPS
4. Streaming WPS
5. Current Status

Lake Analyzer

Lake Analyzer

- Developed at University of Wisconsin/USGS
- Analysis of high-frequency lake buoys data
- Written in Matlab (R version with less features exist)

→ Wind speed

→ Water temperature

→ Schmidt stability

→ (Parent) buoyancy frequency

→ (Parent) lake number

→ (Parent) Mode one vertical seiche period

→ (Parent) metalimnion bottom depth

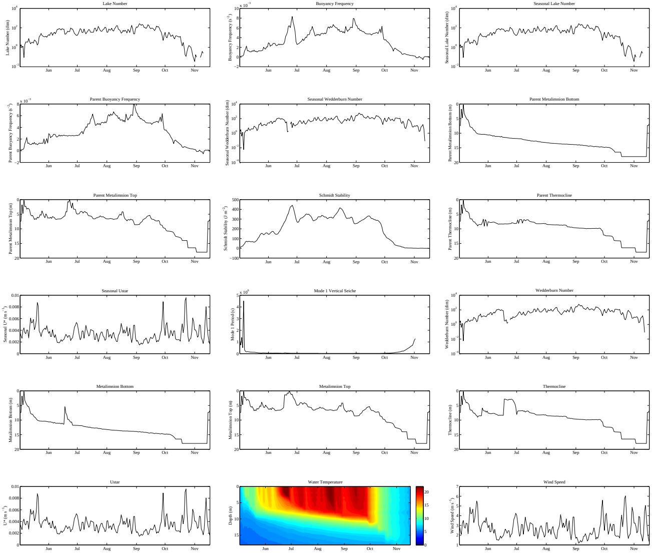
→ (Parent) metalimnion top depth

→ (Parent) thermocline depth

→ (Parent) u star (turblent velocity scale from wind)

→ (Parent) Wedderburn number

Lake Analyzer



Lake Analyzer

- How to to integreate Lake Analyzer into web-processing chains?
- How to offer live analysis of real-time data?
- How to analyze hundreds, thousands, or millions of lakes?

Matlab WPS

Matlab WPS

- Offering Matlab functions as WPS processes
- Implemented as a 52°N WPS backend (and standalone version)
- Configuration with simple YAML file
- Similar approach to WPS4R, but…
  - uses seperated configuration file for each script
  - complex inputs/outputs are handled in Matlab

  → tradeoff: no output conversion between different formats

Matlab WPS — Example

```
function result = add(a, b)
  result = a + b
end
```

```
---
function: add
connection: local
identifier: matlab.add
version: 1.0.0
inputs:
  - identifier: a
    type: double
  - identifier: b
    type: double
outputs:
  - identifier: result
    type: double
...
```

```
<?xml version="1.0" encoding="UTF-8"?>
<ProcessDescription xmlns:wps="[...]"
  xmlns:ows="[...]" wps:processVersion="1.0.0">
  <ows:Identifier>matlab.add</ows:Identifier>
  <ows:Title>matlab.add</ows:Title>
  <DataInputs>
    <Input minOccurs="1" maxOccurs="1">
      <ows:Identifier>a</ows:Identifier>
      <ows:Title>a</ows:Title>
      <LiteralData>
        <ows:DataType ows:reference="xs:double"/>
        <ows:AnyValue/>
      </LiteralData>
    </Input>
    <Input minOccurs="1" maxOccurs="1">
      <ows:Identifier>b</ows:Identifier>
      <ows:Title>b</ows:Title>
      <LiteralData>
        <ows:DataType ows:reference="xs:double"/>
        <ows:AnyValue/>
      </LiteralData>
    </Input>
  </DataInputs>
  <ProcessOutputs>
    <Output>
      <ows:Identifier>result</ows:Identifier>
      <ows:Title>result</ows:Title>
      <LiteralOutput>
        <ows:DataType ows:reference="xs:double"/>
      </LiteralOutput>
    </Output>
  </ProcessOutputs>
</ProcessDescription>
```

Matlab WPS — Websockets

```
GET /streaming HTTP/1.1
Host: example.com
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Key:
       dGhlIHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Version: 13
```

```
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Accept:
       s3pPLMBiTxaQ9kYGzzhZRbK+xOo=
```

Full-Duplex TCP connection

Widely supported:

- IE >10
- Firefox >6
- Chrome >14
- Safari >6
- Opera >12.1

Lake Analyzer WPS

Lake Analyzer WPS

- simple implementation using the *Matlab WPS*
- small modifications to the script to allow file transfers
- wrapper function to get rid of configuration files

Lake Analyzer WPS — Wrapper Function

```
function [results, resultsWtr, StFig, uStFig, LnFig, WFig, wTempFig, wndSpdFig, ...
          metaTFig, metaBFig, thermDFig, SthermDFig, SmetaBFig, SmetaTFig,      ...
          SuStFig, SLnFig, SWFig, N2Fig, SN2Fig, T1Fig, ST1Fig ]                ...
          = Run_LA_WPS(bthFileName, lvlFileName, wndFileName, wtrFileName,      ...
                       salFileName, outputResolution, totalDepth, windHeight,   ...
                       windAveraging, layerAveraging, outlierWindow,            ...
                       maxWaterTemp, minWaterTemp, maxWindSpeed, minWindSpeed,  ...
                       metaMinSlope, mixedTempDifferential, figRes, figUnits,   ...
                       figWidth, figHeight, leftMargin, rightMargin, topMargin, ...
                       botMargin, fontName, fontSize, heatMapMin, heatMapMax)
```

Lake Analyzer WPS — Configuration

```
---
connection:
  host: localhost
  port: 7000
identifier: org.gleon.LakeAnalyzer
version: 1.0.0
title: Lake Analyzer
abstract: Lake Analyzer
function: Run_LA_WPS
inputs:
  # input files
  - identifier: bathymetry
    title: Bathymetry
    abstract: >
      A bathymetry file is a comma delimited (after ver. 3.5, tab delimited)
      text file with extension of [.bth]. The file starts from one line header
      and followed by the hypsographic data at each depth (Example 2.1). Depths
      must start from zero (i.e. surface) with a unit of meters, and
      hypsographic curve data with area as square meters is followed by comma
      delimiter. If the hypsographic curve is not concluded with zero at the
      bottom, LakeAnalyzer program automatically assigns zero to the bottom
      depth which was defined during the configuration process (see section 3).
      LakeAnalyzer linearly interpolates the given hypsographic curve. Change
      to the hypsographic curve due to surface elevation change is not supported
      by the current version of the LakeAnalyzer.
    type: { mimeType: text/csv }
  - identifier: waterLevel
    title: Water Level
    abstract: >
      The Water Level file is a tab delimited text file with the file extension
      of [.lvl]. Water level input is optional for all the outputs. It is useful
      for estuaries and lake with significant level changes which affect
      hypsographic curve of the water body. If the program locates the water
      level file in the correct directory with correct file name, the effect of
      water level fluctuation to the bathymetry area are calculated when
      calculating stabilities. The water level file contains one header
      [DateTime level(positive Z down)]. From the second line, date/time
      information with the format of [yyyy-mm-dd HH:MM], and water level from
      the highest elevation area measurement available (original depth is the
```

Lake Analyzer WPS — Process Description

```
<ProcessDescription xmlns:wps="http://www.opengis.net/wps/1.0.0" xmlns:ows="http://www.opengis.net
       /ows/1.1" statusSupported="false" storeSupported="true" wps:processVersion="1.0.0">
  <ows:Identifier>org.gleon.LakeAnalyzer</ows:Identifier>
  <ows:Title>Lake Analyzer</ows:Title>
  <ows:Abstract>Lake Analyzer</ows:Abstract>
  <DataInputs>
    <Input minOccurs="1" maxOccurs="1">
      <ows:Identifier>bathymetry</ows:Identifier>
      <ows:Title>Bathymetry</ows:Title>
      <ows:Abstract>A bathymetry file is a comma delimited (after ver. 3.5, tab delimited) text
             file with extension of [.bth]. The file starts from one line header and followed by
             the hypsographic data at each depth (Example 2.1). Depths must start from zero (i.e.
             surface) with a unit of meters, and hypsographic curve data with area as square
             meters is followed by comma delimiter. If the hypsographic curve is not concluded
             with zero at the bottom, LakeAnalyzer program automatically assigns zero to the
             bottom depth which was defined during the configuration process (see section 3).
             LakeAnalyzer linearly interpolates the given hypsographic curve. Change to the
             hypsographic curve due to surface elevation change is not supported by the current
             version of the LakeAnalyzer.</ows:Abstract>
      <ComplexData>
        <Default>
          <Format>
            <MimeType>text/csv</MimeType>
          </Format>
        </Default>
        <Supported>
          <Format>
            <MimeType>text/csv</MimeType>
          </Format>
        </Supported>
      </ComplexData>
    </Input>
    <Input minOccurs="1" maxOccurs="1">
      <ows:Identifier>waterLevel</ows:Identifier>
      <ows:Title>Water Level</ows:Title>
      <ows:Abstract>The Water Level file is a tab delimited text file with the file extension of
             [.lvl]. Water level input is optional for all the outputs. It is useful for estuaries
              and lake with significant level changes which affect hypsographic curve of the water
              body. If the program locates the water level file in the correct directory with
```

Lake Analyzer WPS — Process Description

```
<Output>
  <ows:Identifier>results</ows:Identifier>
  <ows:Title>Raw Results</ows:Title>
  <ComplexOutput>
    <Default>
      <Format>
        <MimeType>text/csv</MimeType>
      </Format>
    </Default>
    <Supported>
      <Format>
        <MimeType>text/csv</MimeType>
      </Format>
    </Supported>
  </ComplexOutput>
</Output>
<Output>
  <ows:Identifier>N2</ows:Identifier>
  <ows:Title>Buoyancy frequency</ows:Title>
  <ComplexOutput>
    <Default>
      <Format>
        <MimeType>image/png</MimeType>
        <Encoding>Base64</Encoding>
      </Format>
    </Default>
    <Supported>
      <Format>
        <MimeType>image/png</MimeType>
        <Encoding>Base64</Encoding>
      </Format>
    </Supported>
  </ComplexOutput>
</Output>
```

Lake Analyzer WPS — Example Request

```
<wps:Execute xmlns:wps="http://www.opengis.net/wps/1.0.0" xmlns:ows="http://www.opengis.net/ows
      /1.1" xmlns:xlink="http://www.w3.org/1999/xlink" service="WPS" version="1.0.0">
 <ows:Identifier>org.gleon.LakeAnalyzer</ows:Identifier>
 <wps:DataInputs>
   <wps:Input>
     <ows:Identifier>bathymetry</ows:Identifier>
     <wps:Reference method="GET" mimeType="text/csv" xlink:href="http://localhost/example.bth" />
   </wps:Input>
   <wps:Input>
     <ows:Identifier>waterLevel</ows:Identifier>
     <wps:Reference method="GET" mimeType="text/csv" xlink:href="http://localhost/example.lvl" />
   </wps:Input>
   <wps:Input>
     <ows:Identifier>windSpeed</ows:Identifier>
     <wps:Reference method="GET" mimeType="text/csv" xlink:href="http://localhost/example.wnd" />
   </wps:Input>
   <wps:Input>
     <ows:Identifier>waterTemperature</ows:Identifier>
     <wps:Reference method="GET" mimeType="text/csv" xlink:href="http://localhost/example.wtr" />
   </wps:Input>
   <wps:Input>
     <ows:Identifier>outputResolution</ows:Identifier>
     <wps:Data>
       <wps:LiteralData dataType="xs:int">86400</wps:LiteralData>
     </wps:Data>
   </wps:Input>
   <wps:Input>
     <ows:Identifier>totalDepth</ows:Identifier>
     <wps:Data>
       <wps:LiteralData dataType="xs:double">20.0</wps:LiteralData>
     </wps:Data>
   </wps:Input>
   <wps:Input>
     <ows:Identifier>windHeight</ows:Identifier>
     <wps:Data>
       <wps:LiteralData dataType="xs:double">2.0</wps:LiteralData>
     </wps:Data>
   </wps:Input>
   <wps:Input>
```

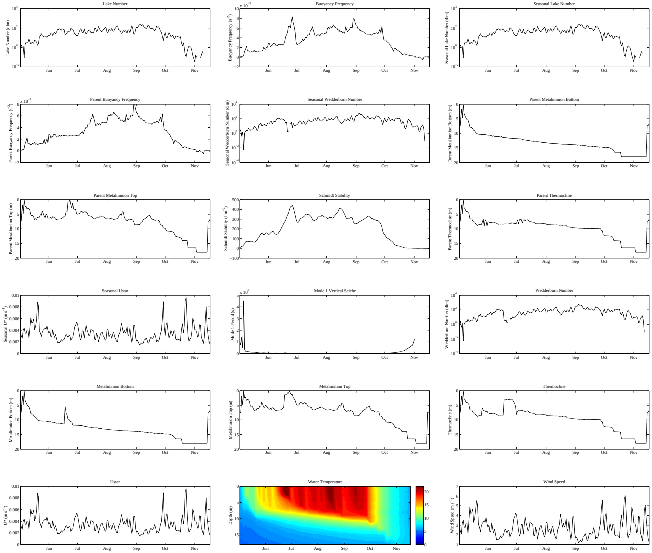
Lake Analyzer WPS — Example Response

```xml
<wps:ExecuteResponse xmlns:wps="http://www.opengis.net/wps/1.0.0" xmlns:ows="http://www.opengis.
      net/ows/1.1" serviceInstance="http://localhost:12121/WebProcessingService?REQUEST=
      GetCapabilities&amp;SERVICE=WPS" xml:lang="en-US" service="WPS" version="1.0.0">
  <wps:Process wps:processVersion="1.0.0">
    <ows:Identifier>org.gleon.LakeAnalyzer</ows:Identifier>
    <ows:Title>Lake Analyzer</ows:Title>
  </wps:Process>
  <wps:Status creationTime="2014-01-26T17:35:59.908+01:00">
    <wps:ProcessSucceeded>Process successful</wps:ProcessSucceeded>
  </wps:Status>
  <wps:ProcessOutputs>
    <wps:Output>
      <ows:Identifier>results</ows:Identifier>
      <ows:Title>Raw Results</ows:Title>
      <wps:Data>
        <wps:ComplexData mimeType="text/csv">
          DateTime  St  uSt Ln  W wndSpd  metaT metaB thermD  SthermD SmetaB  SmetaT  SuSt  SLn SW
                N2  SN2 T1  ST1
          2009-05-02 00:00  8.3074  0.0036283 0.91411 1.5148  2.8906  7.5171  7.5171  7.5171
                7.5171  7.5171  7.5171  0.0036283 0.91411 1.5148  9.3472e-05  9.3472e-05
                1037167.3004  1037167.3004
          2009-05-03 00:00  11.7715 0.0030324 1.8307  2.9882  2.4159  7.4102  7.4102  7.4102
                7.4102  7.4102  7.4102  0.0030324 1.8307  2.9882  0.00010163  0.00010163
                736824.197  736824.197
          2009-05-04 00:00  19.7603 0.0026169 1.0003  0.39255 2.0848  1.7917  1.7917  1.7917
                1.7917  1.7917  1.7917  0.0026169 1.0003  0.39255 0.00020563  0.00020563
                1381887.1043  1381887.1043
          2009-05-05 00:00  22.6096 0.0040067 1.2942  1.3588  3.192 4.7376  4.7376  4.7376  4.7376
                  4.7376  4.7376  0.0040067 1.2942  1.3588  0.00019596  0.00019596  512954.1566
                512954.1566
          2009-05-06 00:00  35.6483 0.0044006 0.089499  0.0056531 3.5056  0.25  0.25  0.25  0.25
                0.25  0.25  0.0044006 0.089499  0.0056531 0.00042287  0.00042287  4507642.1775
                4507642.1775
```

Lake Analyzer WPS — Example Response

```xml
<wps:Output>
  <ows:Identifier>N2</ows:Identifier>
  <ows:Title>Buoyancy frequency</ows:Title>
  <wps:Data>
    <wps:ComplexData encoding="Base64" mimeType="image/png">
      iVBORw0KGgoAAAANSUhEUgAAAaEAAADSCAIAAABsC+hOAAAACXBIWXMAAC4jAAAuIwF4pT
      92AAAAB3RJTUUH3gEaECQcH+tXLQAAACR0RVh0U29mdHdhcmUATUFUTEFCLCBUaGUgTWF0
      aFdvcmtzLCBJbmMuPFjdGAAAACJ0RVh0Q3JlYXRpb24gVGltZQAyNi1KYW4tMjAxNCAxNz
      ozNjoyOHM2gTwAAAriSURBVHic7d3RlqOqFgXQJKP+/5dzHjxtWZoYo4Cb7ZwPd9Tpm65G
      gSUCmvvz+bwBJPU4uwAAFck4IDMZB2Qm44DMZByQmYzjqPv9/vJniEDG0Yj44xQyDsjs5+
      wCkNM4ahs2mQ//Ofzvu018+Znn8zn7DBwk4yhvSKvpz0N4zZJr9pmXP8BBMo7ypsOxlaha
      ztCNHxZwlCLjqOLjcGw21mtXMi7GmgPlTSfalvk1+xMBR1vmPShgtnrw8k9Ycg9ub9Ycxqk6c3
      DUoFUBmblXBTKbZ5zJESCTh4cNgcQe48qXGV8gn9/9cfsCztAPqO3I8OvQmoOAA4Ir8JxD
      /Dvcjh7z7qioN6WtpqOi3iqX9vhAyt4RILOHpQYgsbfPRW8Jvr5G1EB3jofM0XVVgMi8Ww
      n49fJlCl2z5gBkJuOAzGQckJmMAzKTccD/Uj6dKeOAzGQckJmMAzKTcUBmMg7ITMYBmXle
      FULI96BoEMZxQGbRx3EubsARxnFAZjIOyKybjEv5JB1QWzcZB7QxzH2nGVXIOCAzGQdk1l
      PGpRk8A830lHEA3/qzB9iGWyCZx5hr9/v9+Y+7QiCHh0SD0+mD9RR4XtUdLlBWwdD/zbjp
      gC5aWhlsQm71wuc344b5uOXPH0ULRGCHIWWCdOdpFh38VfaOAJk9vhqyAfTlZwy4yPNxcB
      2GHWX9WVd1ZoFkzMcBmck4IDMZB2Qm44DMZByQmYwDTlb1SU0ZB2QWOuM8hwbl/xgwl+
      mxog4yziQFsFsHGQewm4wDMpNxQGZ/vkPaPh0gmceYa/f7/flPjvUUIL7aafMZNps4NZg
      HGfMCF3rpQv/fP7IJ70cKtCLgoM783FAZn8yboy25d3rCoEI1NFj8G55Q
      EyeXw1ZAPoy88YcNMtI1IPLmXo+/s6fvBx0uv5uPbcogI1nDMfJ9HgpUwvNQri/DUHgHpk
      HPC1I/N3jZ2ccSsj845OIuxmoW9UcE/cVKBxXKUjBK6swPOqFGHcyksGegdFGcepP6CGEB
      kn4Ebu05nSHo47/15VwN005Qm3ZpQVYhzHYFxlFnk0UOQtGPE3Lcs4CCpycHmeE+fndCuPM91qdvagGXdcpz
      3/4rHeXX316+OprnHbdEr9ps24XhixQlUnZJwNUBBK7uHz+XuAa+ilzgzi9nGZZLtY96o1
      mmwveTfTabFLybqGTnuxMo5vSYGsjPFLiZtxavdb8q47brobKJ9x527a6HG/CF9Z1m+n+4
      Si2Z6zszFm8Ld5xx3H7ZCjlbukFxGhdxHBi4w7khShUiZ4+y54roIf6bl6fxaqhvYXgI21
      UKOyHmkq3nW7dyanqOHxfD4rPdbX8mnBdwF38S0IDSaq7hNlf/PF665HMSvrzx7g+/0+i7
      wtPn5+fPHAluvz8Wv4t1OnRf7R4CoNctu06fVqitmvyhp75cZOdKKCJSxVs03XHCqNKbq+
      S21QbBNS+VhK3u4343YH8POf6Z8UKBoLu5v1WEG994r161nvR3fQjuAr1VWnIVCqFmapst
      ufcdx4gorPytVLva4HcV8ZK2X2w3Dsy+a1rMTeY67fkjfQcvq7L49x+DZN4pY3UCx9DKOD
      jVgfSENVfvSomjWVKmDHaHy5/Hdu4+ioaX68/Tlxvq+j07hi1gc/dskcR93MizWHyCOsyG
      VLqf0lQRV3LeB8SLt11XrvTdIrBi+b15GTs95Sv2rHu5dKNv5hjyrtRWWmWMY122uW5jXz
      V7MjMa+zgXGfgIOmgGqN4yJvJemxTZw11VVk/b5B4aO1se1WSr7joHbMjS73fiWT6r0jV3
      Bwgn/WlKNtJU3c0+L4Kv6Wn689QVR8CatAxr0rSu3ltu3neuVKlbVTxYmtIyWZ7WfqaLhR
      5OIx/ettnoMsYrqT4bZaqjZzEXXHcX21y1A2TrW83HZwPOBets4dv3bWiAsmr0Y1cB4+cq
```

Lake Analyzer WPS — Example Response



Streaming WPS

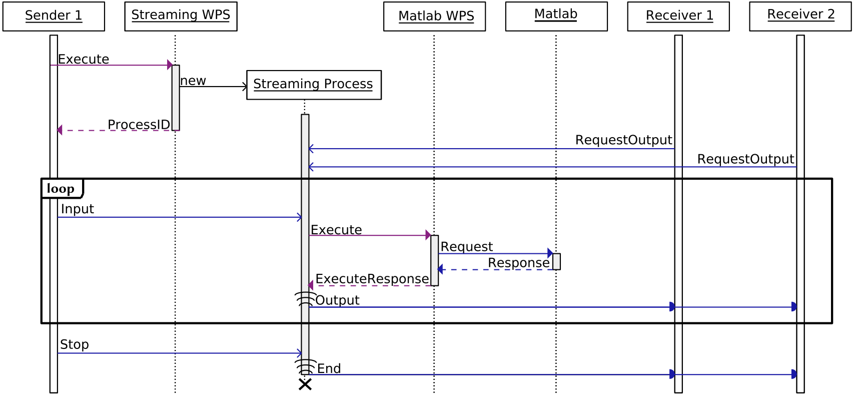
Streaming WPS

- Allows streaming of of inputs and outputs to a WPS process
- Previous approach:
    - WPS is splitting inputs
    - publishing results in a playlist that has to be checked constantly
- This approach:
    - Sender splits inputs
    - Sender starts a WPS streaming process
    - Receiver connects to the WPS process
    - Sender sends small chunks to the streaming process
    - Streaming process pushes the results to the receiver

Streaming WPS

- In a web-based (i.e. browser-based) environment, the WPS interface
  - … does not allow subsequent inputs
  - … does not allow intermediate outputs
- Only possible with input/output references and long-lasting requests/polling
- Solution: break out of the WPS interface
  - Start a *background* process
  - Stream inputs/outputs using *WebSockets*

Streaming WPS — Sequence Diagram



Streaming WPS — Input Types

- Streaming Inputs
  - submitted with `<stream:InputMessage>`
  - of type `<wps:Input>`
- Static Inputs
  - submitted with initial `<wps:Execute>`
  - merged with inputs of every streaming iteration
  - of type `<wps:Input>`
- Reference Inputs
  - submitted with `<stream:InputMessage>`
  - references the output of a previous or upcoming streaming iteration
- Polling Inputs

Streaming WPS — Handling Dependencies

- Clients declare dependencies to other streaming iterations (or their outputs)
- Automatic declaration of (spatial) dependencies not possibe as it is use case and format specific
- Process waits for all dependencies to become available
- Checking for cyclic dependencies/execution ordering
    - → dynamic topological sort algorithm for directed acyclic graphs (based on breadth first search)

Streaming WPS — Handling Dependencies

```
<stream:ReferenceInput>
  <ows:Identifier>input3</ows:Identifier>
  <stream:Reference>
    <wsa:MessageID>uuid:f31da315-bce3-4e26-8112-3ccf0ecf1ab5</wsa:MessageID>
    <stream:Output>output1</stream:Output>
  </stream:Reference>
</stream:ReferenceInput>
```

```
<wsa:RelatesTo RelationshipType="https://github.com/autermann/streaming-wps/needs">
      uuid:f31da315-bce3-4e26-8112-3ccf0ecf1ab5</wsa:RelatesTo>
```

Current Status

Current Status

- Done:
  - ✓ Matlab WPS
  - ✓ Lake Analyzer WPS
- WIP:
  - ✗ Streaming WPS
- Upcoming:
  - ✗ Webapp showcasing the process chain
- Sources:
  - → https://github.com/autermann/Lake-Analyzer
  - → https://github.com/autermann/matlab-connector
  - → https://github.com/autermann/matlab-wps
  - → https://github.com/autermann/streaming-wps

Thanks. Questions?