OCTOBER 24, 2023

Building User-specific Dashboards from Legacy Views with AVEVA™ PI Vision

Dave Johnson

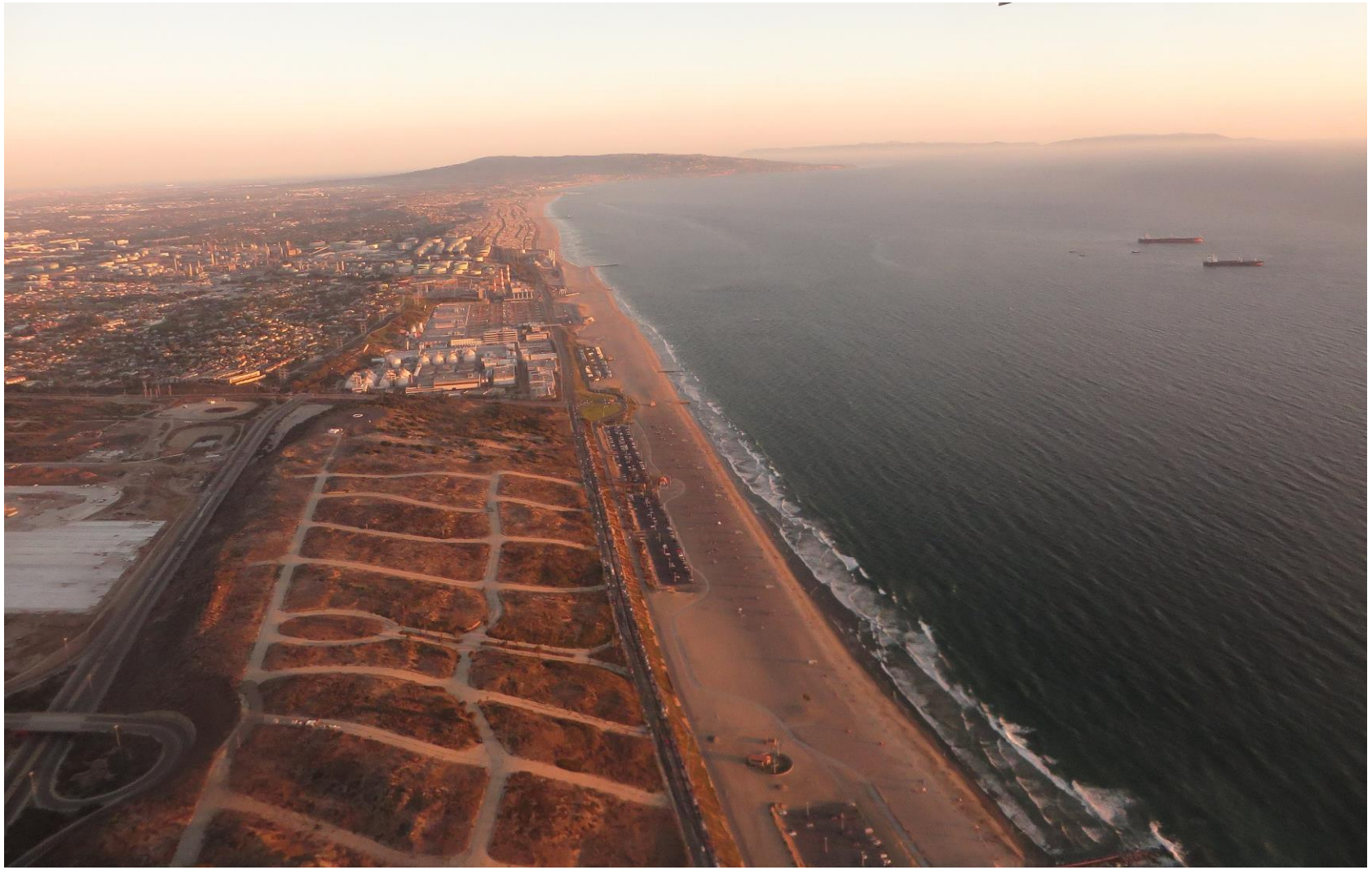
Senior Software Engineer, Chevron

Agenda

- Intro
- Challenge: Migrate thousands of displays from a legacy system quickly to PI Vision
- Solution: Use the PI Vision API to programmatically build PI Vision displays
- Benefits
- **PI Vision API Bonus tips**
- Questions

AVEVA

Intro

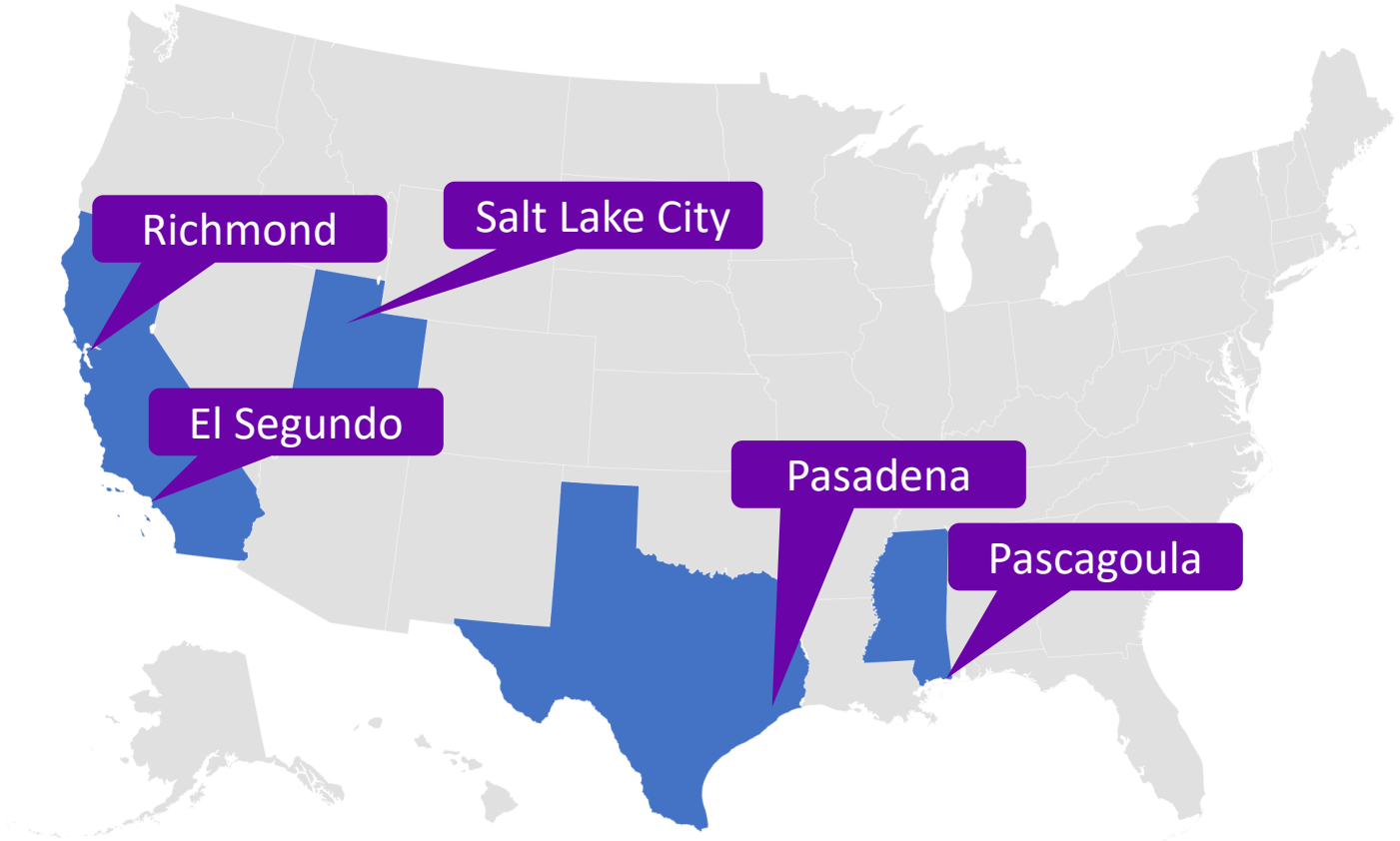


Chevron

AVEVA

Chevron Domestic Refineries

I serve in a technical advisory role for all Chevron refineries and liaison with our central PI team

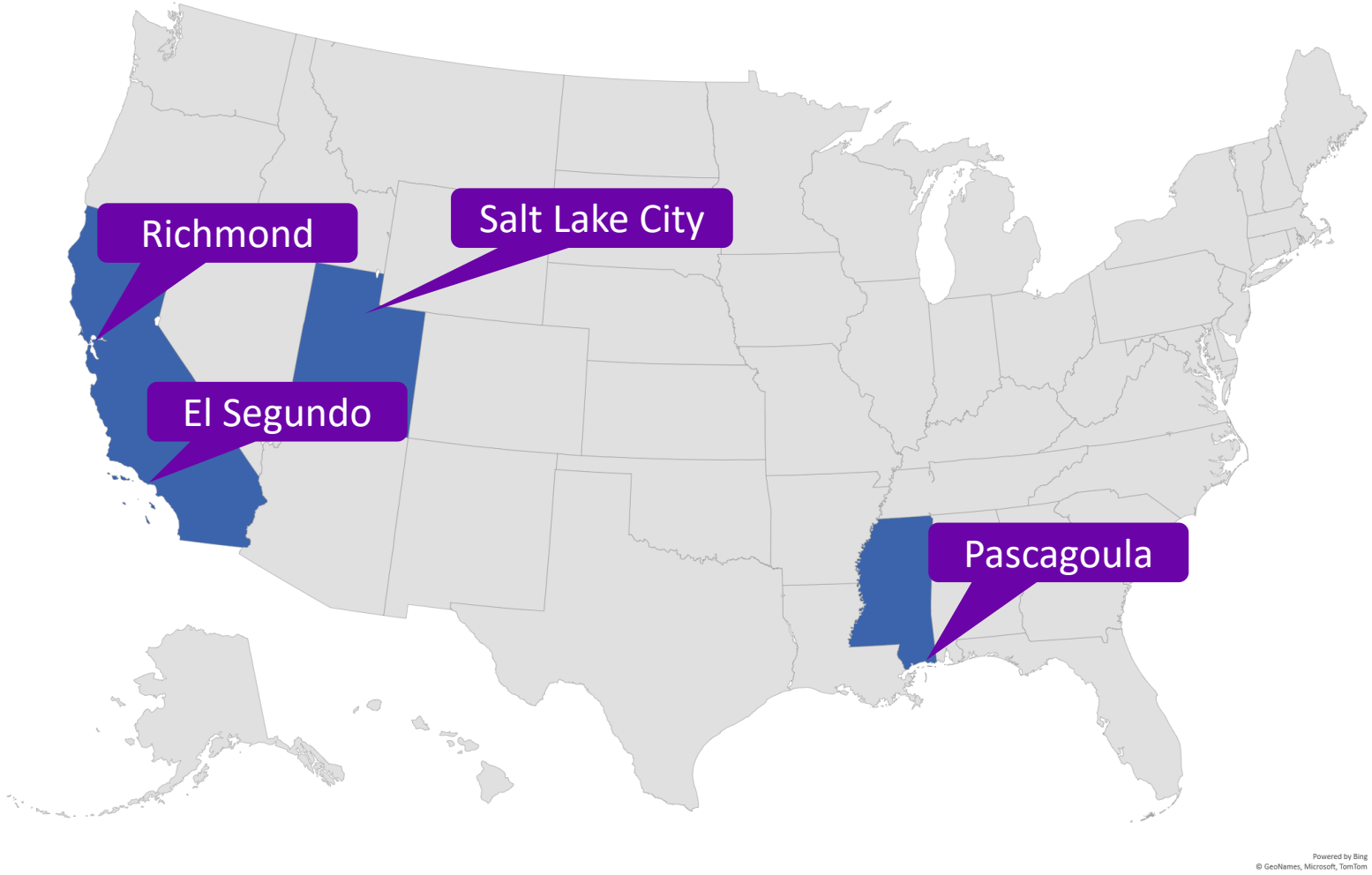


Legacy Visualization = “LVIZ”

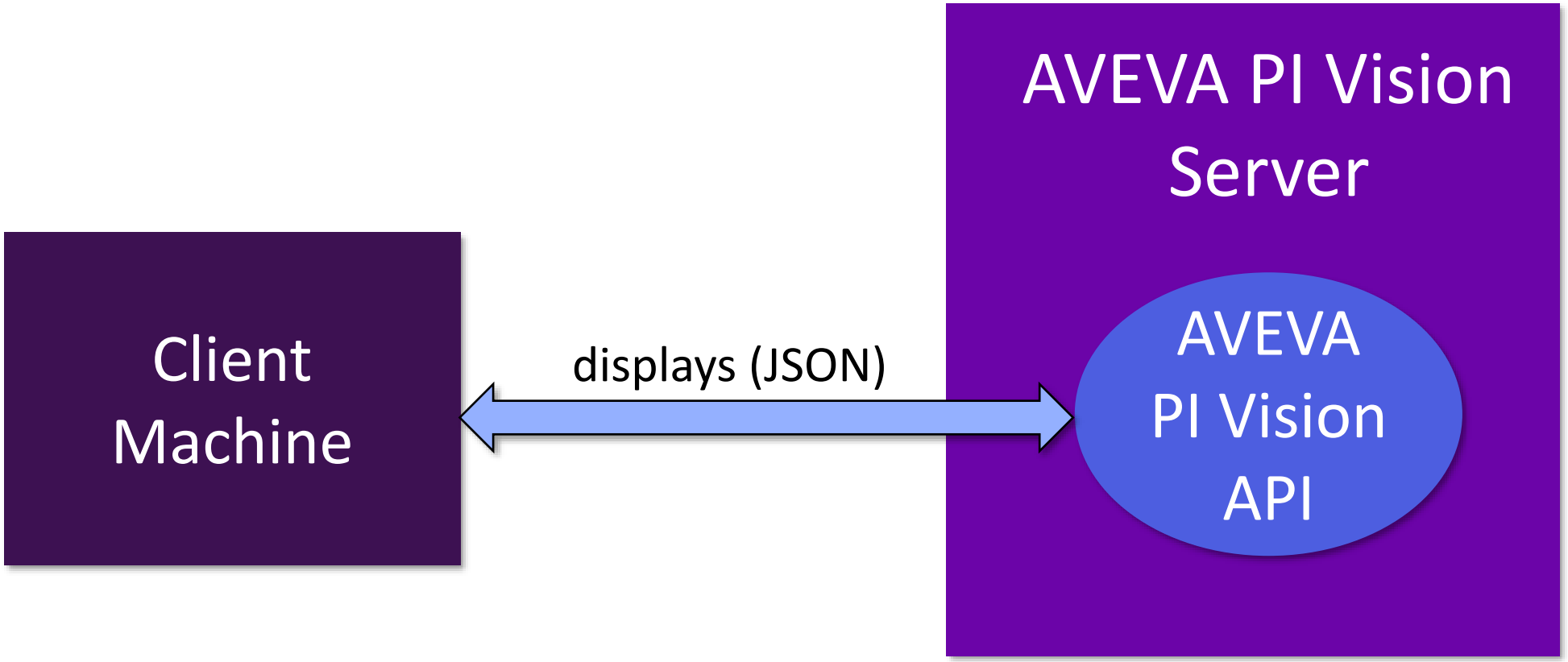
LVIZ =

Elvis Deployments

Challenge: Migrate thousands of displays from a legacy system **fraught with security vulnerabilities** to PI Vision



PI Vision API to the rescue!



Nota Bene

PI Vision API ≠ PI Web API

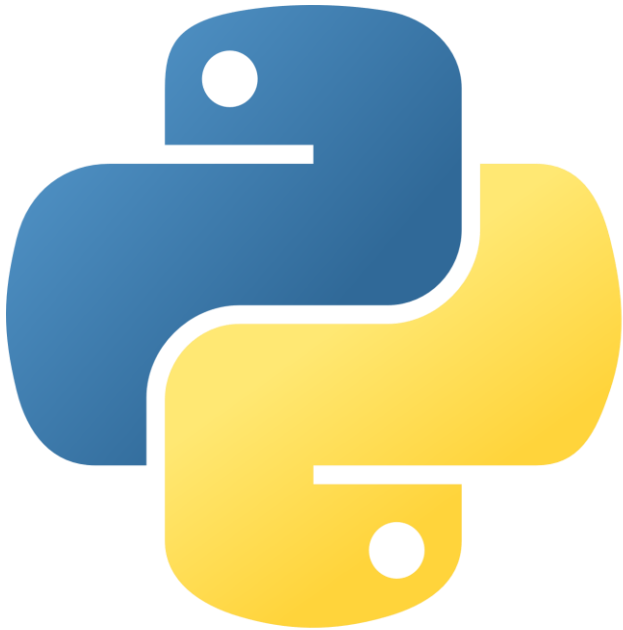
Important Note

With experienced PI administrators and internal development resources, Chevron was able to go beyond the supported capabilities of AVEVA™ PI Vision and create a customized solution to meet their specific needs. This presentation celebrates Chevron's achievement in going beyond what is available out-of-the-box and is not intended to imply or warrant the functionality they describe for general usage of AVEVA™ PI Vision.

AVEVA

PI Vision API Client Options

- C#
- cURL
- Golang
- PowerShell
- Python
- Rust
- Many other options – just need to be able to consume (and publish) JSON data from an HTTP-based endpoint.

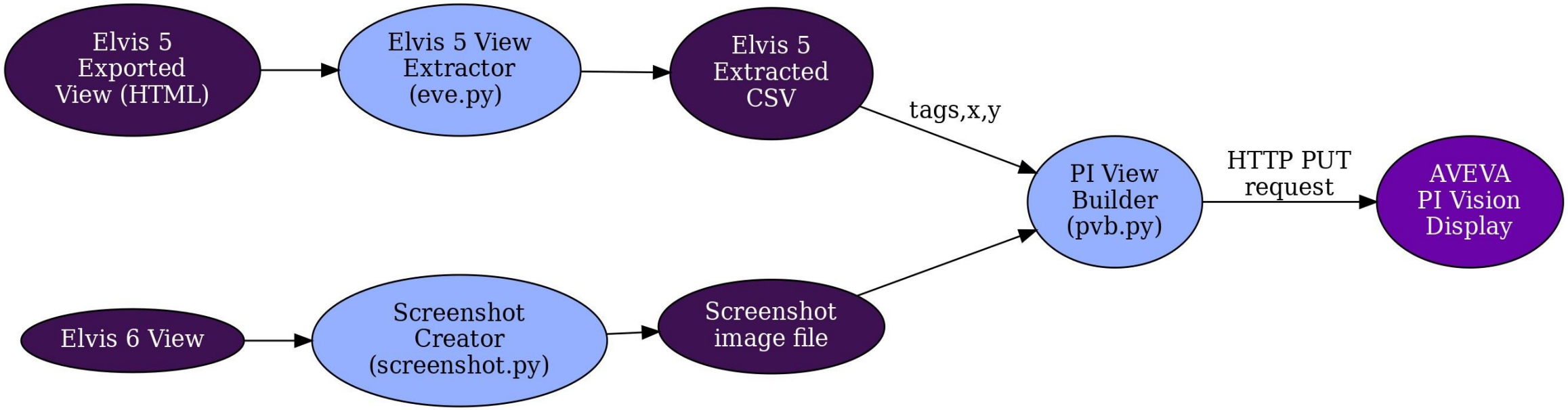


- The PI Vision API opens many new doors of possibility in the realm of PI Vision!

AVEVA

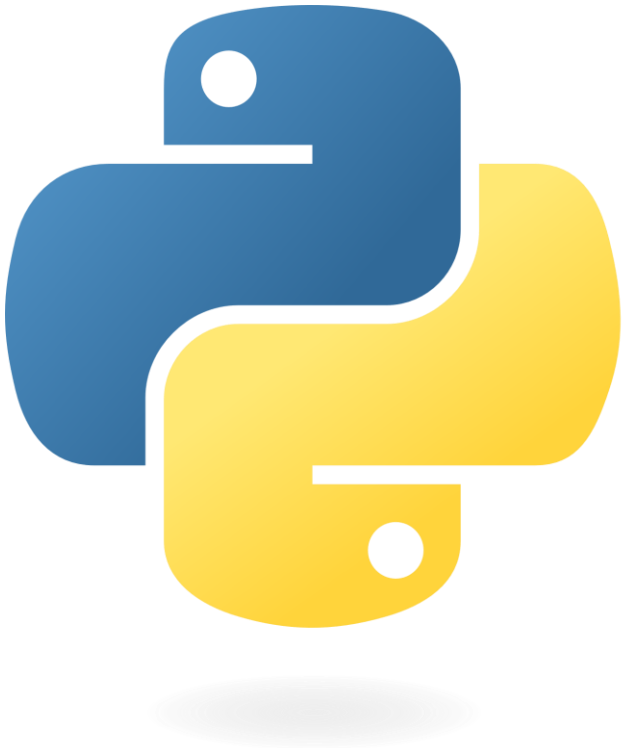
High Level Architecture Overview

Phase 1: **Pure PI tag-based conversion** rather than leveraging PI AF capabilities



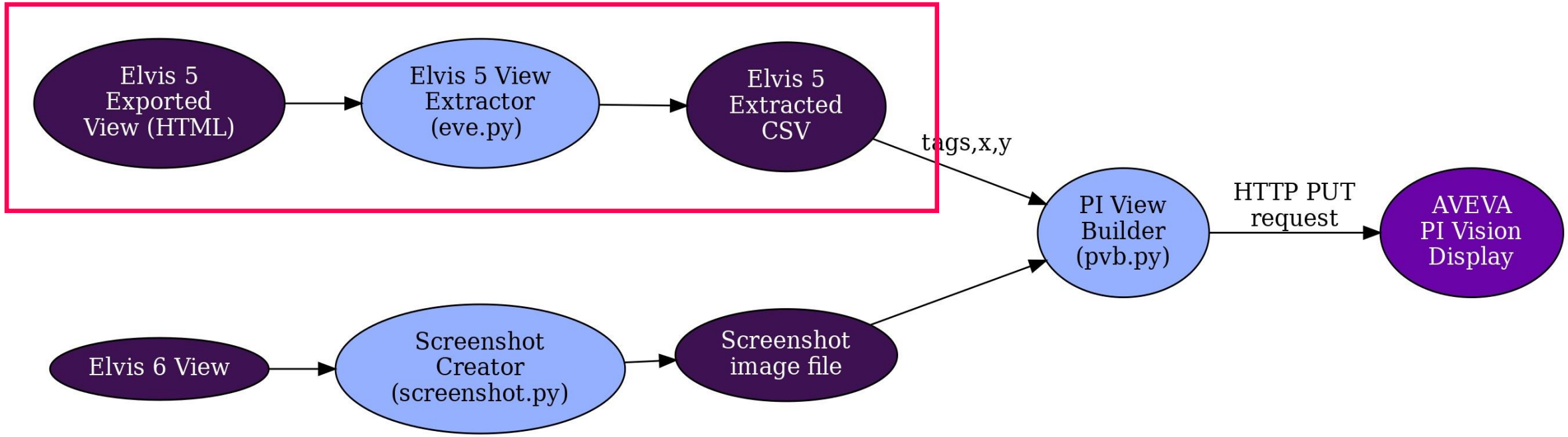
Key Python Packages Used

- pythonnet (Python/C# interop) to invoke .NET libraries
  - PI AF SDK
  - .NET WebClient to get and post JSON data (could not use the Python *requests* module due to authentication challenges)
- BeautifulSoup4 (bs4)
- cssutils
- pyautogui
- Pillow



AVEVA

High Level Architecture Overview



Elvis 5 Exported View (Fragment)

18. Template at (677,554)

Template			
Component	View	Drillable	Template Settings
Controller_TS	Name: HorizValve Device: Desktop Orientation: Landscape	Use View Properties	Component View List Member: BaseTag Expression: "22FC101"

19. Template at (44,511)

Template			
Component	View	Drillable	Template Settings
Controller_TS	Name: HorizValve Device: Desktop Orientation: Landscape	Use View Properties	Component View List Member: BaseTag Expression: "22FC105"

Elvis 5 Exported HTML (Fragment)

BeautifulSoup web scraping saves the day!

```
<tr>
  <td width="3%"><br />18.</td>
  <td><br />Template at (677,554)</td>
</tr>
<tr>
  <td></td>
  <td>
    <table
      border
      cellspacing="0"
      cellpadding="0"
      width="100%"
      HEIGHT="50%"
    >
      <tr bgcolor="#FFFFCC">
        <td colspan="4">
          <center>Template</center>
        </td>
      </tr>
      <tr bgcolor="#FFFFCC">
        <td width="20%"><center>Component</center></td>
        <td width="30%"><center>View</center></td>
        <td width="10%"><center>Drillable</center></td>
        <td width="40%"><center>Template Settings</center></td>
      </tr>
```

```
<tr>
  <td>Controller_TS</td>
  <td>
    Name: HorizValve<br />
    Device: Desktop<br />
    Orientation: Landscape<br />
  </td>
  <td>Use View Properties</td>
  <td>
    <table
      border
      cellspacing="0"
      cellpadding="0"
      COLS="2"
      width="100%"
    >
      <tr bgcolor="#CCCCCC">
        <td colspan="2" width="30%">
          <center>Component View List</center>
        </td>
      </tr>
      <tr bgcolor="#CCCCCC">
        <td width="10%"><center>Member</center></td>
        <td width="20%"><center>Expression</center></td>
      </tr>
      <tr>
        <td>BaseTag</td>
        <td>"22FC101"</td>
      </tr>
    </table>
```


AVEVA

Elvis View Extractor (eve.py)

```
C\..> python eve.py -h
usage: eve.py [-h] [-p] [-n] site html_file_or_directory

Extract PI tags and their X/Y coordinates into a CSV file from the HTML files created during the Elvis 5 view export process

positional arguments:
  site                  Name of the site. Sites are retrieved from the sites.csv config file. Valid options include: ese,pas,ric,slc
  html_file_or_directory
                        HTML file to process or directory of HTML files to process all at once

options:
  -h, --help            show this help message and exit
  -p, --prod            use the prod PI server rather than the dev PI server to fetch PI tag values
  -n, --nopi            bypass fetching PI metadata from server
```

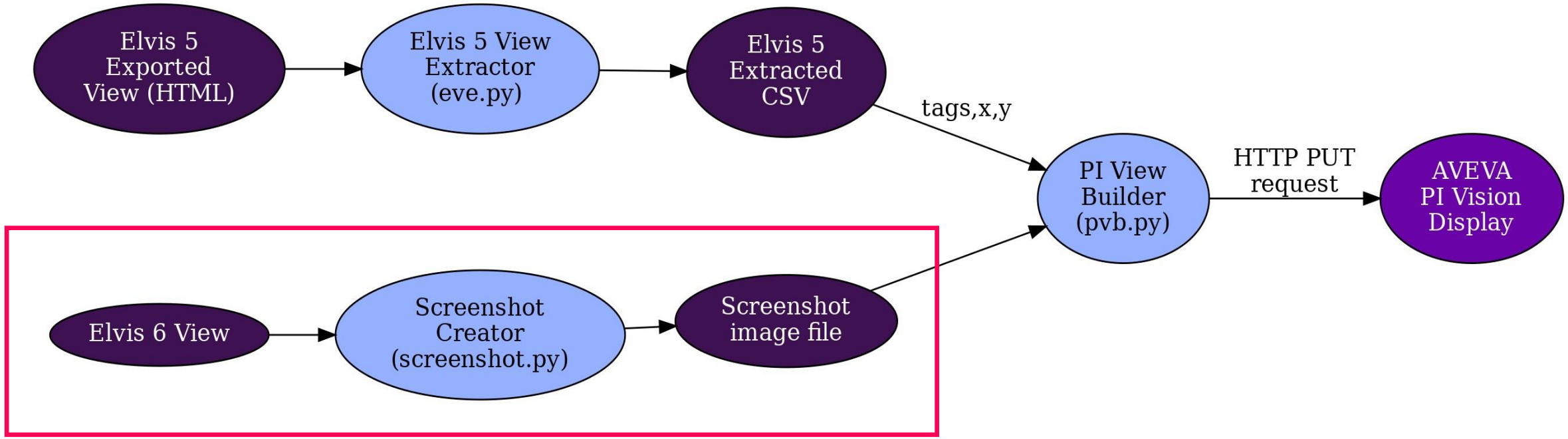

AVEVA

Elvis 5 Exported View CSV result

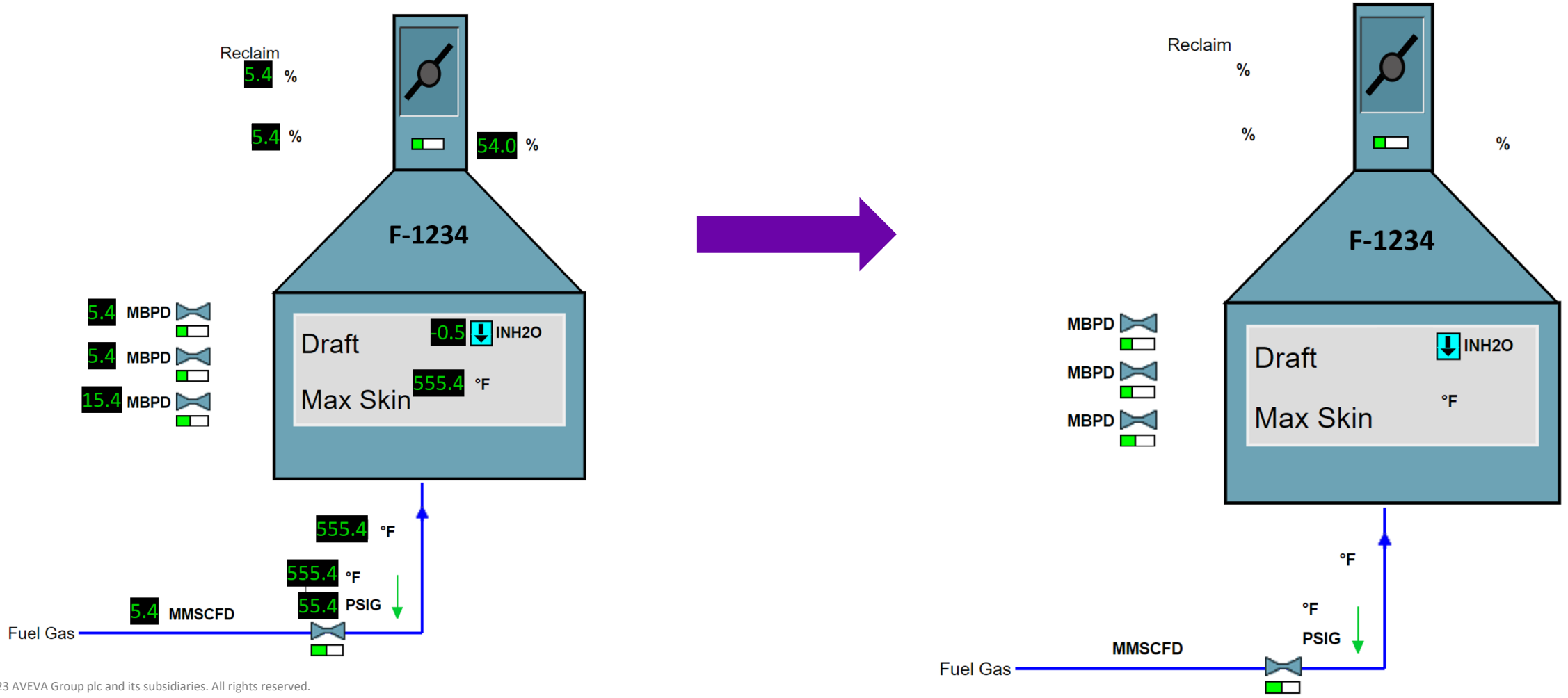
	A	B	C	D	E	F	G	H	I	J
1	**Display Name**	**Row**	**X**	**Y**	**Symbol**	**Tag Name**	**Description**	**UOM**	**PointType**	**Extended Props**
2	FCC	18	677	554	HorizValve	22FC101	Description 1	MBPD	Float32	BaseTag:22FC101
3	FCC	19	44	511	HorizValve	22FC105	Description 2	LB/HR	Float32	BaseTag:22FC105
4	FCC	20	207	215	HorizValve	22LC450	Description 3	PCT	Float32	BaseTag:22LC450
5	FCC	24	359	393	VertBar30_PV	22LC460	Description 4	PCT	Float32	BaseTag:22LC460
6	FCC	27	391	240	Description_11pt	22TI4614	Description 5	DEGF	Float32	
7	FCC	28	455	237	TagUnit	22TI4614	Description 6	DEGF	Float32	
8	FCC	29	391	261	Description_11pt	22TI4609	Description 7	DEGF	Float32	
9	FCC	30	455	258	TagUnit	22TI4609	Description 8	DEGF	Float32	
10	FCC	31	353	191	TagUnit	22TI460	Description 9		Float32	
11	FCC	32	391	282	Description_11pt	22TI4615	Description 10	DEGF	Float32	
12	FCC	33	455	279	TagUnit	22TI4615	Description 11	DEGF	Float32	
13	FCC	34	391	302	Description_11pt	22TI4616	Description 12	DEGF	Float32	
14	FCC	35	391	322	Description_11pt	22TI4617	Description 13	DEGF	Float32	
15	FCC	36	455	300	TagUnit	22TI4616	Description 14	DEGF	Float32	

AVEVA

High Level Architecture Overview



Screenshot element zapping

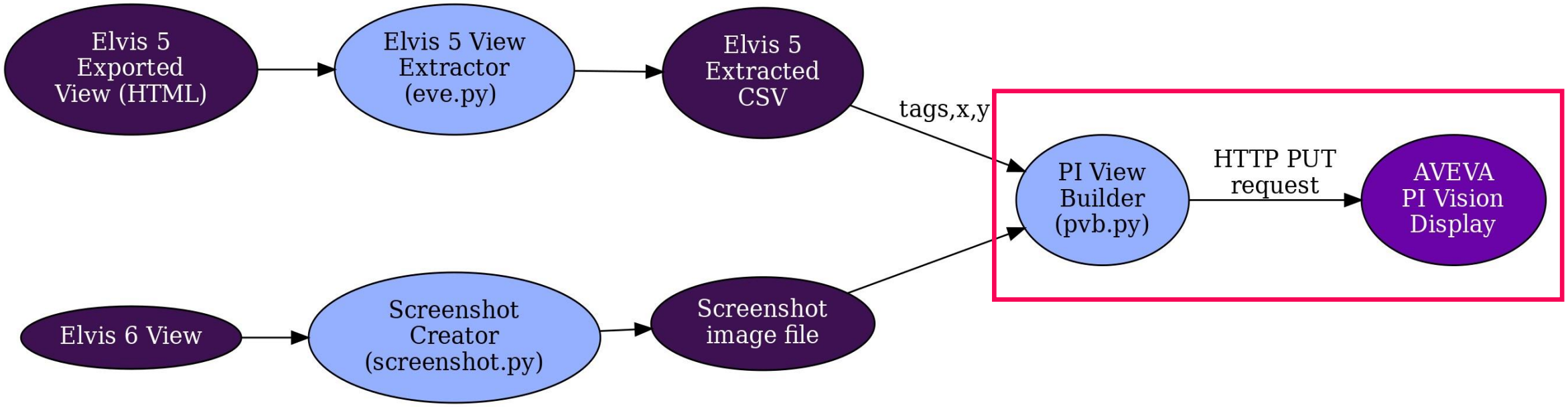


Screenshot Creator (screenshot.py)

Result: Image without live elements

```
C\...>python screenshot.py -h
usage: screenshot.py [-h] [-f] [-z] site group

Create background images for PI Vision displays from Elvis 6 HTML5 views

positional arguments:
  site                  Name of the site. Sites are retrieved from the sites.csv config file. Valid options include: ese,pas,ric,slc
  group                 Group to process. Separate multiple groups with commas and no spaces. Example: fcc1,fcc2

options:
  -h, --help            show this help message and exit
  -f, --force-elvis6    Force an update of the offline files from the Elvis 6 server even if the files already exist
  -z, --zap             Zap elements manually from the display before saving a screenshot
```

High Level Architecture Overview



Elvis 5 Exported View CSV result

	A	B	C	D	E	F	G	H	I	J
1	**Display Name**	**Row**	**X**	**Y**	**Symbol**	**Tag Name**	**Description**	**UOM**	**PointType**	**Extended Props**
2	FCC	18	677	554	HorizValve	22FC101	Description 1	MBPD	Float32	BaseTag:22FC101
3	FCC	19	44	511	HorizValve	22FC105	Description 2	LB/HR	Float32	BaseTag:22FC105
4	FCC	20	207	215	HorizValve	22LC450	Description 3	PCT	Float32	BaseTag:22LC450
5	FCC	24	359	393	VertBar30_PV	22LC460	Description 4	PCT	Float32	BaseTag:22LC460
6	FCC	27	391	240	Description_11pt	22TI4614	Description 5	DEGF	Float32	
7	FCC	28	455	237	TagUnit	22TI4614	Description 6	DEGF	Float32	
8	FCC	29	391	261	Description_11pt	22TI4609	Description 7	DEGF	Float32	
9	FCC	30	455	258	TagUnit	22TI4609	Description 8	DEGF	Float32	
10	FCC	31	353	191	TagUnit	22TI460	Description 9		Float32	
11	FCC	32	391	282	Description_11pt	22TI4615	Description 10	DEGF	Float32	
12	FCC	33	455	279	TagUnit	22TI4615	Description 11	DEGF	Float32	
13	FCC	34	391	302	Description_11pt	22TI4616	Description 12	DEGF	Float32	
14	FCC	35	391	322	Description_11pt	22TI4617	Description 13	DEGF	Float32	
15	FCC	36	455	300	TagUnit	22TI4616	Description 14	DEGF	Float32	

AVEVA

PI Vision JSON symbol templates

Mapped to symbols defined in the CSV

```
{
    "DataSources": [
        "pi:\\\\{{pi_server}}\\{{tag}}"
    ],
    "Name": "Symbol35",
    "SymbolType": "horizontalgauge",
    "Configuration": {
        "DataShape": "Gauge",
        "Height": 11.197997404893931,
        "Width": 24.862637362637262,
        "Fill": "#22b14c",
        "Background": "#ffffff",
        "Stroke": "#414853",
        "StrokeWidth": 2,
        "ValueStroke": "#414853",
        "ShowLabel": false,
        "ShowValue": false,
        "ShowUOM": true,
        "Top": 70.87,
        "Left": 809.72,
        "FormatType": "F1",
        "ValueScaleSettings": {
            "MinType": 2,
            "MinValue": -30,
            "MaxType": 2,
            "MaxValue": 100
        },
        "ValueScale": false
    }
}
```

AVEVA

PI Vision JSON symbol templates

Dynamic Expression Syntax

```
51      {
52          "DataSources": [
53              "pi:\\\\{{pi_server}}\\{{tag1}}.op"
54          ],
55          "Name": "Symbol30",
56          "SymbolType": "horizontalgauge",
57          "Configuration": {
58              "DataShape": "Gauge",
59              "Height": 10.1123595505617774,
60              "Width": 23.595505617977636,
61              "Fill": "#7fff00",
62              "Background": "#ffffff",
63              "Stroke": "#000000",
64              "StrokeWidth": 2,
65              "ValueStroke": "black",
66              "ShowLabel": false,
67              "ShowValue": false,
68              "ShowUOM": false,
69              "Top": "{{y+50}}",
70              "Left": "{{x-18}}",
71              "ValueScale": false,
72              "LinkURL": ""
73          }
74      },
```

AVEVA

PI View Builder (pvb.py)

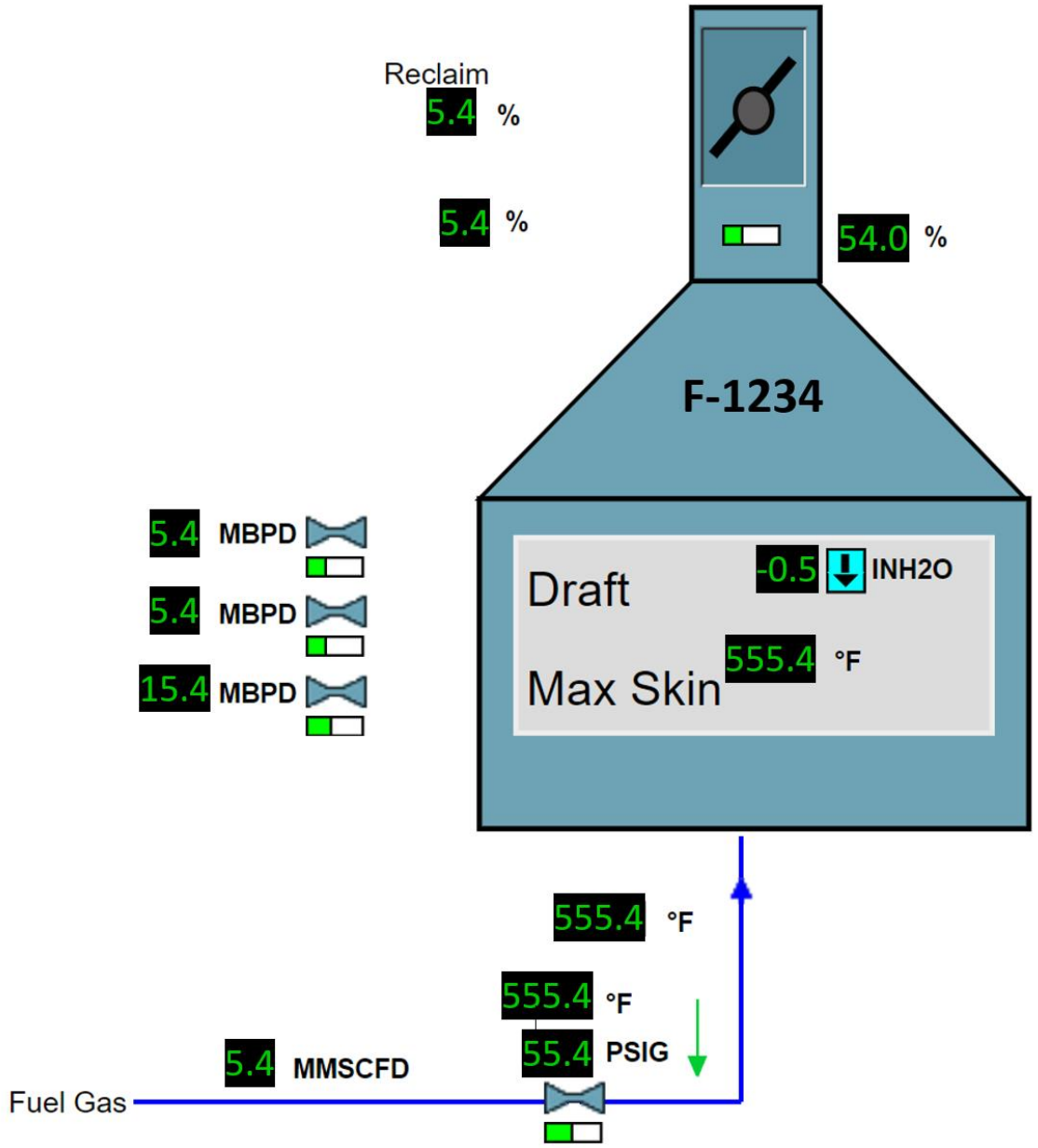
Result: PI Vision Display of legacy converted display including background screenshot

```
C\...>python pvb.py -h
usage: pvb.py [-h] [-u] [-p] [-b] site group

Build PI Vision displays using CSV files and screenshots created using eve.py and screenshot.py respectively

positional arguments:
  site           Name of the site. Sites are retrieved from the sites.csv config file. Valid options include: ese,pas,ric,slc
  group          Group to process. Separate multiple groups with commas and no spaces. Example: fcc1,fcc2

options:
  -h, --help     show this help message and exit
  -u, --update   update the display on the PI Vision web server in addition to writing the results to a JSON file
  -p, --prod     use the prod environment rather than the dev environment
  -b, --browser  launch web browser after display is updated on the PI Vision web server (must be used with --update to work)
```

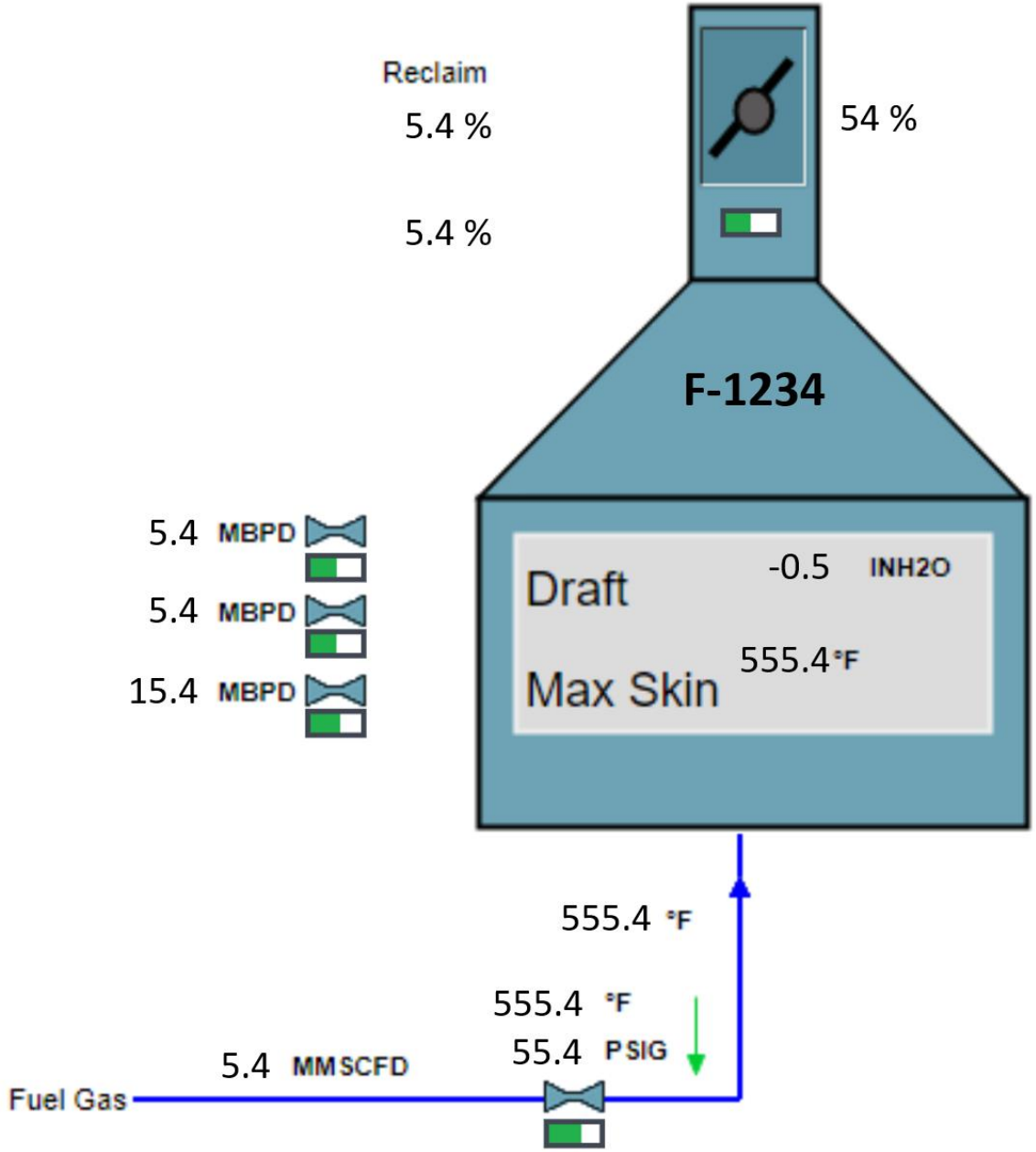
Note: The PI Vision API officially supports just the import and export of displays. We created symbols and exported displays as a starting point to ensure the JSON conformed to standards before programmatically building the PI Vision displays.

AVEVA

Final Result: Elvis graphic



Final Result: AVEVA PI Vision Graphic



Surgeon (surgeon.py)

Perform surgery on an existing PI Vision display rather than regenerating it completely

```
C\...>python surgeon.py -h
usage: surgeon.py [-h] [-u] [-p] [-b] site group

positional arguments:
  site           Name of the site. Sites are retrieved from the sites.csv config file. Valid options include: ese,pas,ric,slc
  group          Group to process. Separate multiple groups with commas and no spaces. Example: fcc1,fcc2

options:
  -h, --help     show this help message and exit
  -u, --update   update the display on the PI Vision web server in addition to writing the results to a JSON file
  -p, --prod     use the prod environment rather than the dev environment
  -b, --browser  launch web browser after display is updated on the PI Vision web server (must be used with --update to work)
```

Oil & Gas | USA

Chevron saves money and increases reliability using the PI Vision API to convert legacy displays

Challenge

- We (Chevron) were faced with a situation in our domestic refineries where we needed to convert hundreds of complex legacy data visualization views built over 20 years residing in a non-AVEVA PI system to AVEVA PI Vision—and we needed to do it expeditiously.

Solution

- We leveraged the AVEVA PI Vision API in conjunction with Python to programmatically build the PI Vision displays rather than creating each new PI Vision display manually.

Results

- **Hundreds of hours saved building new PI Vision displays manually.**
- **Increased accuracy and reliability since the human element of transcribing PI tags (i.e., swivel chair interface) was removed.**
- **We grew our capabilities to use the PI Vision API in other future contexts beyond legacy display conversion.**

PI Vision API Bonus tips

We have just scratched the surface in terms of the flexibility and power the PI Vision API provides

- Create trends or tabular displays automatically from a spreadsheet or simple markdown document
- Find PI tag dependencies
- Count equipment such as pumps, etc. based on symbols present in the views
- Search/replace symbols made across many displays in your solution
- Adjust links in displays after the fact due to a new URL
- **The PI Vision API opens many new doors of possibility in the realm of PI Vision**

AVEVA

Special thanks

- Billy Crosby
- Tom Hosea
- Bryan Klosiewicz
- Brent Bregenzer
- James Owens
- The entire Chevron PI team

Dave Johnson

Senior Software Engineer

- Chevron
- dave.johnson@chevron.com

Questions?

Please wait for the microphone.

State your name and company.

Please remember to…

Navigate to this session in the mobile app to complete the survey.

Thank you!

This presentation may include predictions, estimates, intentions, beliefs and other statements that are or may be construed as being forward-looking. While these forward-looking statements represent our current judgment on what the future holds, they are subject to risks and uncertainties that could result in actual outcomes differing materially from those projected in these statements. No statement contained herein constitutes a commitment by AVEVA to perform any particular action or to deliver any particular product or product features. Readers are cautioned not to place undue reliance on these forward-looking statements, which reflect our opinions only as of the date of this presentation.

The Company shall not be obliged to disclose any revision to these forward-looking statements to reflect events or circumstances occurring after the date on which they are made or to reflect the occurrence of future events.

 linkedin.com/company/aveva

 @avevagroup

ABOUT AVEVA

AVEVA is a world leader in industrial software, providing engineering and operational solutions across multiple industries, including oil and gas, chemical, pharmaceutical, power and utilities, marine, renewables, and food and beverage. Our agnostic and open architecture helps organizations design, build, operate, maintain and optimize the complete lifecycle of complex industrial assets, from production plants and offshore platforms to manufactured consumer goods.

Over 20,000 enterprises in over 100 countries rely on AVEVA to help them deliver life's essentials: safe and reliable energy, food, medicines, infrastructure and more. By connecting people with trusted information and AI-enriched insights, AVEVA enables teams to engineer efficiently and optimize operations, driving growth and sustainability.

Named as one of the world's most innovative companies, AVEVA supports customers with open solutions and the expertise of more than 6,400 employees, 5,000 partners and 5,700 certified developers. The company is headquartered in Cambridge, UK.

Learn more at www.aveva.com

AVEVA