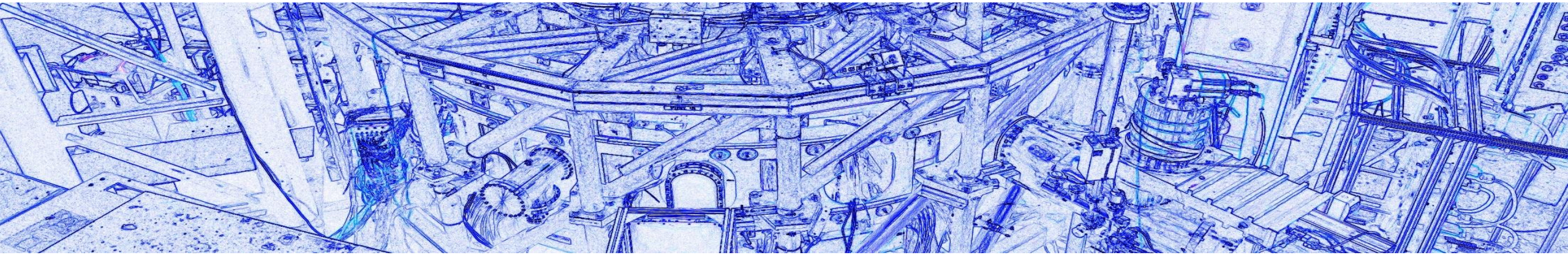
A full stack data acquisition, archive and access solution for J-TEXT based on Web technologies

Yuxing Wang and J-TEXT Team

IAEA TM 2019 Daejeon, Korean

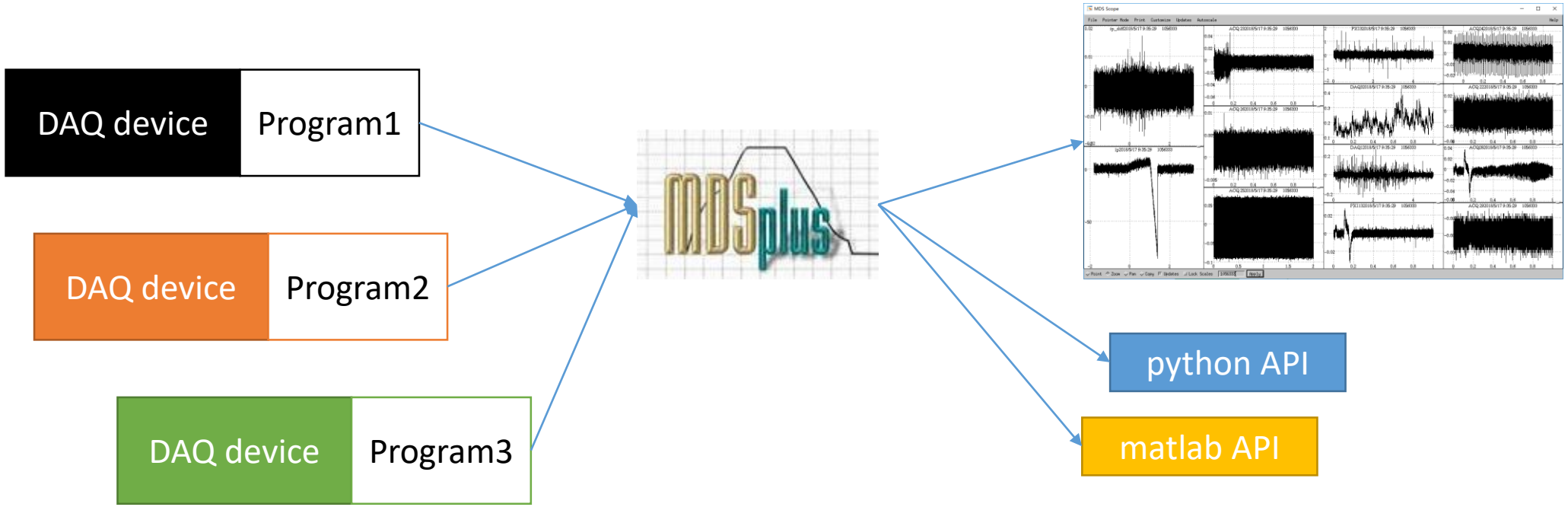


Outline

- Current solution for J-TEXT data acquisition, archive and access system
- The CFET software system framework based on .NET and Web
- Data acquisition system
- Data archive and access system
- User interface and Data Visualization
- Future work

Current solution for J-TEXT

- **Data acquisition** Various DAQ programs
- **Data archive** MDSplus
- **Data access** APIs provided by MDSplus
- **Data visualization** Jscope/dwscope

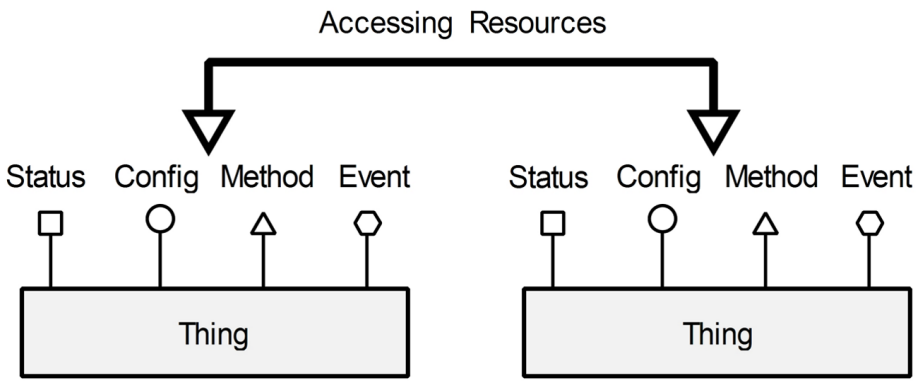


Main issue of current solution

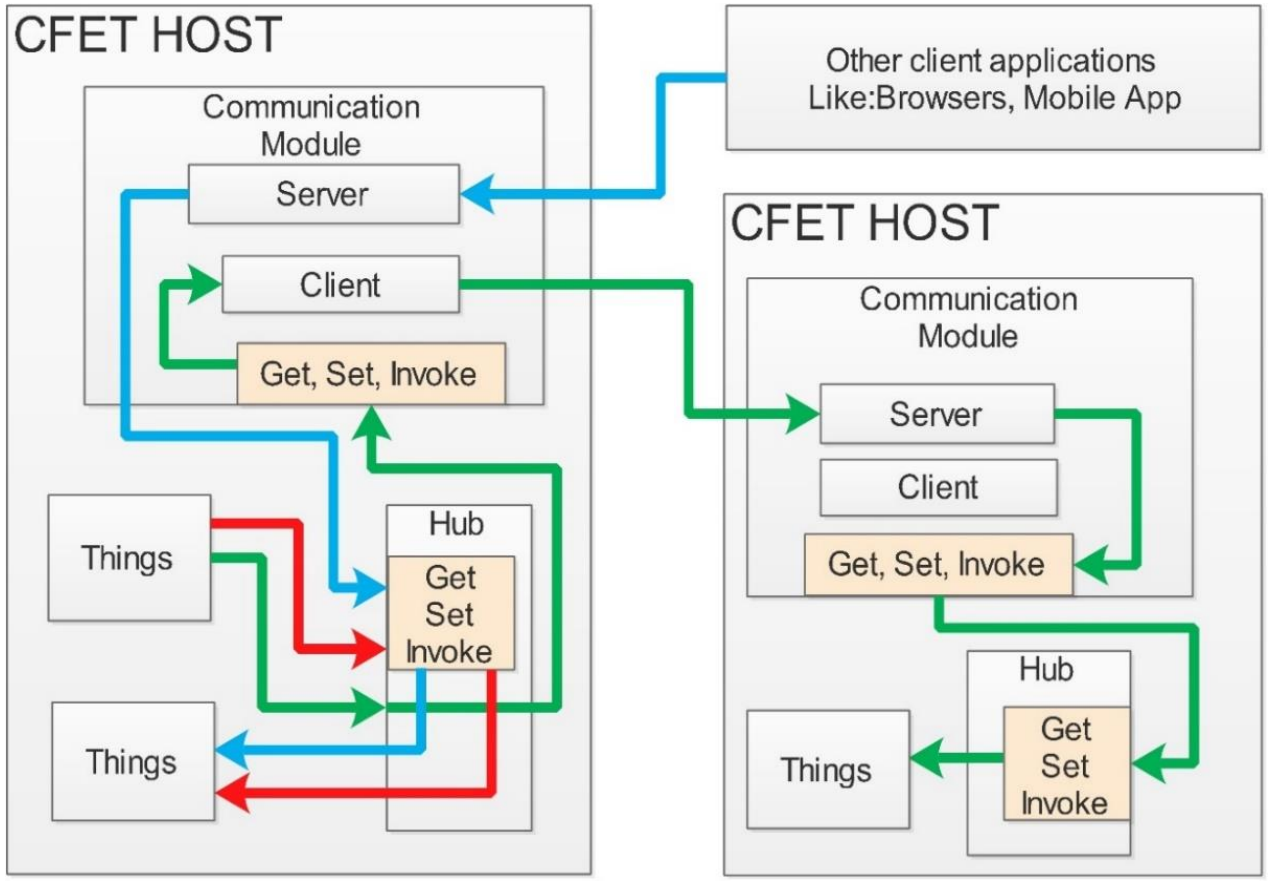
- So many DAQ programs, similar but incompatible, almost no software architecture
- Need installing MDSplus or other software to see the experiment data
- New demands are increasing, and unhappy to modify the existing system

CFET Software system framework

- **Control system Framework for Experimental Devices Toolkit**



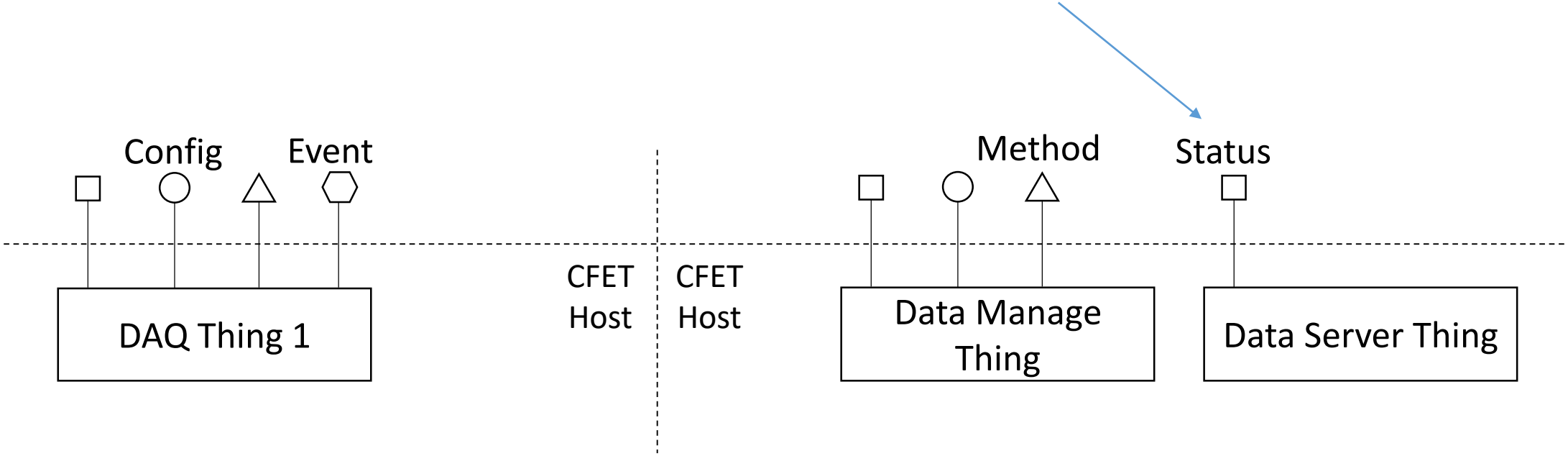
- Thing is something with its own control logic and Resources
- A CFET Host is a program which contains some CFET Things.
- A Thing's Resources can be accessed through the Web by URls, and Things contact each others by their Resources.



Unified Data Access (UDA)

All resources in CFET system can be accessed by an URL through the Web like this：

http://192.168.1.1/tagServer/dataComplex/0/ecei_group1_ch1?start=0&stride=100&count=1000&block=1



A simplest DAQ system use CFET

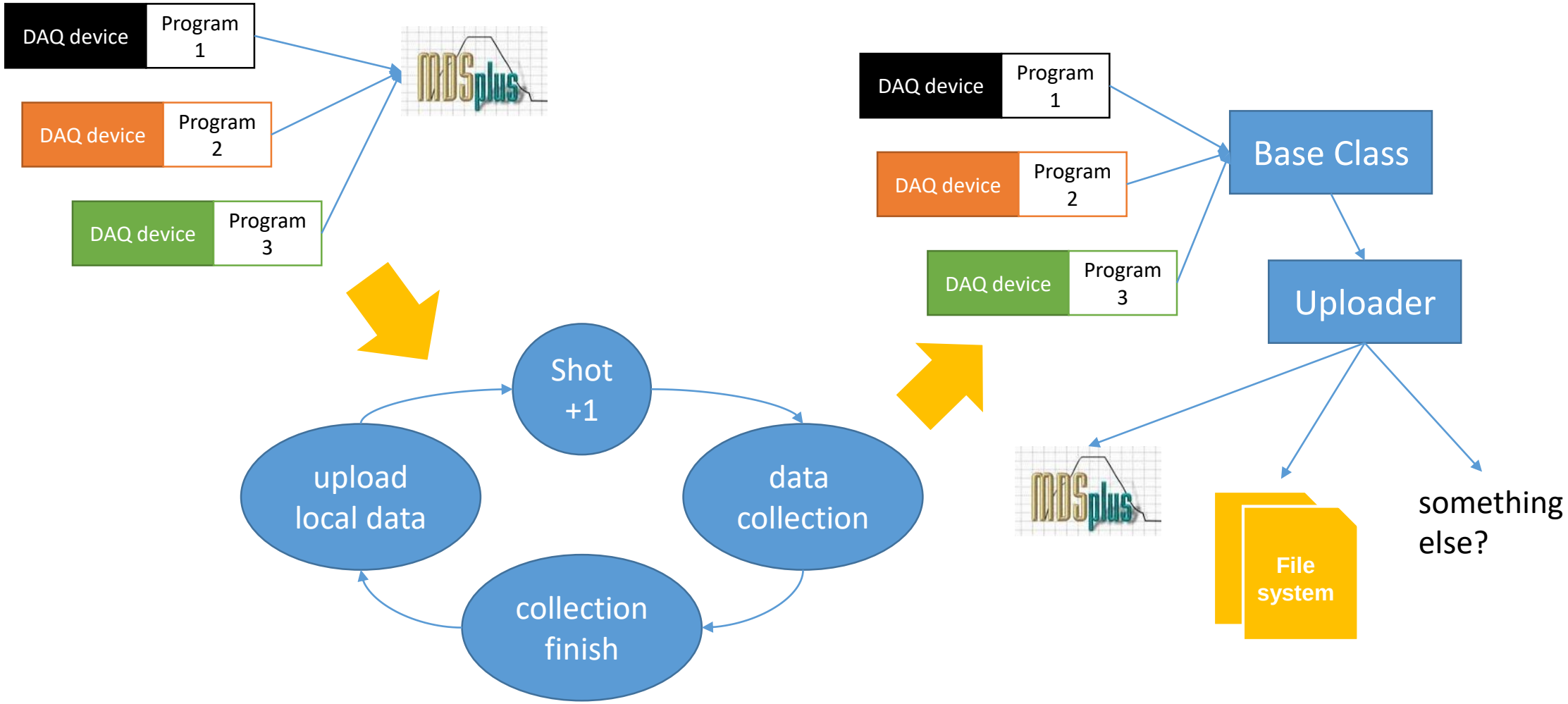
```
public class DaqDevice:Thing
{
    [Cfet2Method]
    public void TryStop()
    {
        MyHub.EventHub.Publish("/DAQ1", "CollectionFinished", null);
    }

    [Cfet2Method]
    public void TryArm()...

    [Cfet2Status]
    public State DaqStatus...
}
```

```
public class Uploader:Thing
{
    public Uploader()
    {
        MyHub.EventHub.Subscribe(new EventFilter("/DAQ1", "CollectionFinished"), handler);
    }

    private void handler(EventArg obj)...

    [Cfet2Status]
    public State UploaderStatus...

    [Cfet2Config]
    public void UploadPath(string dest, string sour)...
}
```

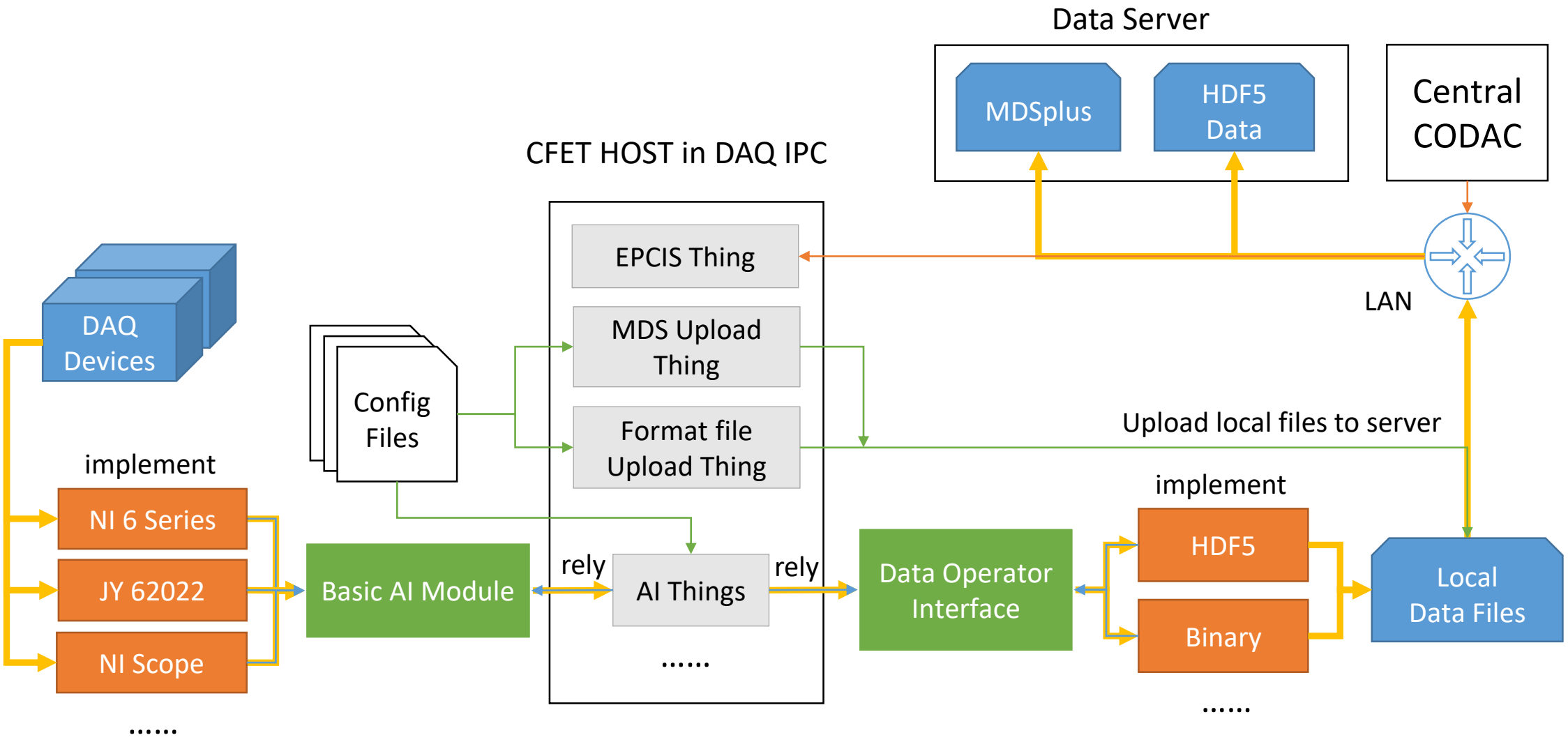
New DAQ and Data Access system

- A flexible architecture to facilitate the addition of new features and later modifications
- Acquisition systems should not rely on specific hardware
- Data storage and archive systems should not depend on a particular database or file format but have supports for them like HDF5
- Data access should meet all existing needs

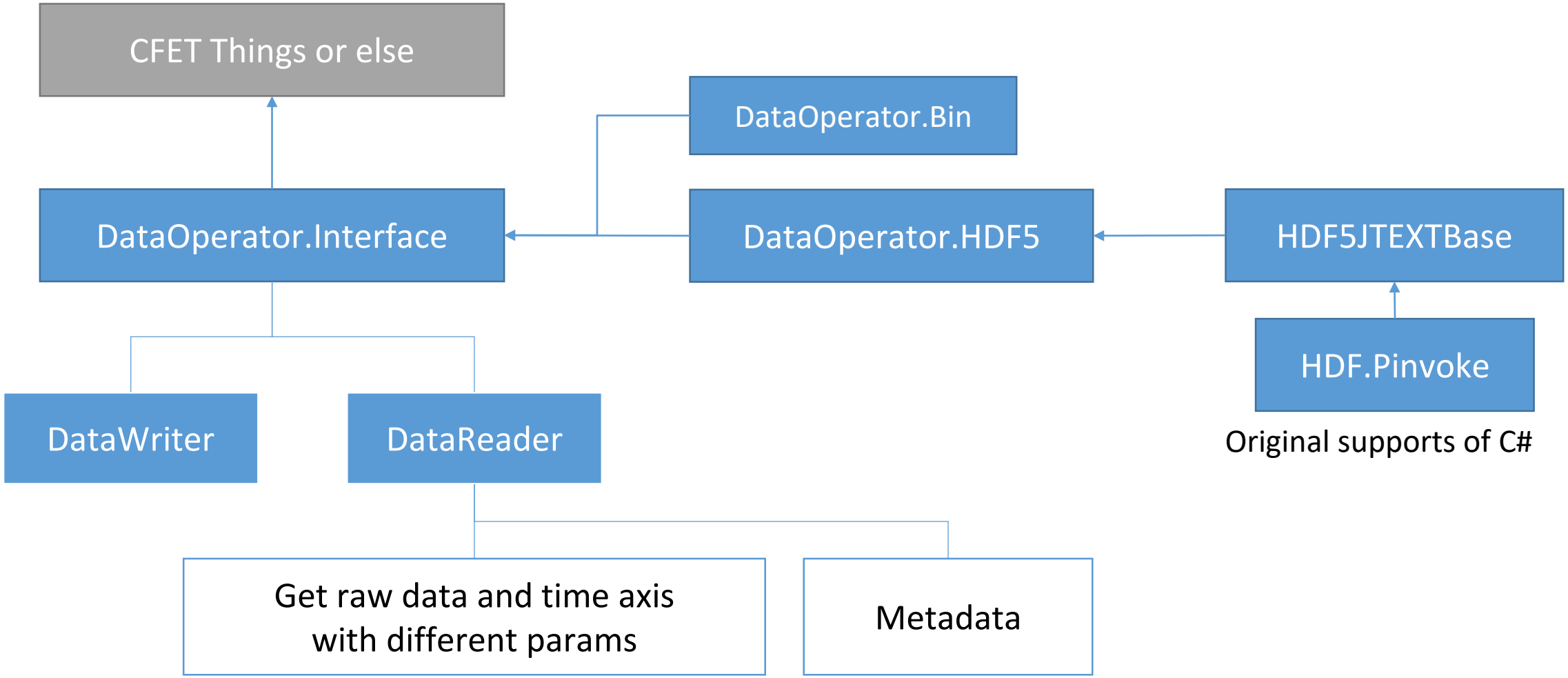
Data acquisition system



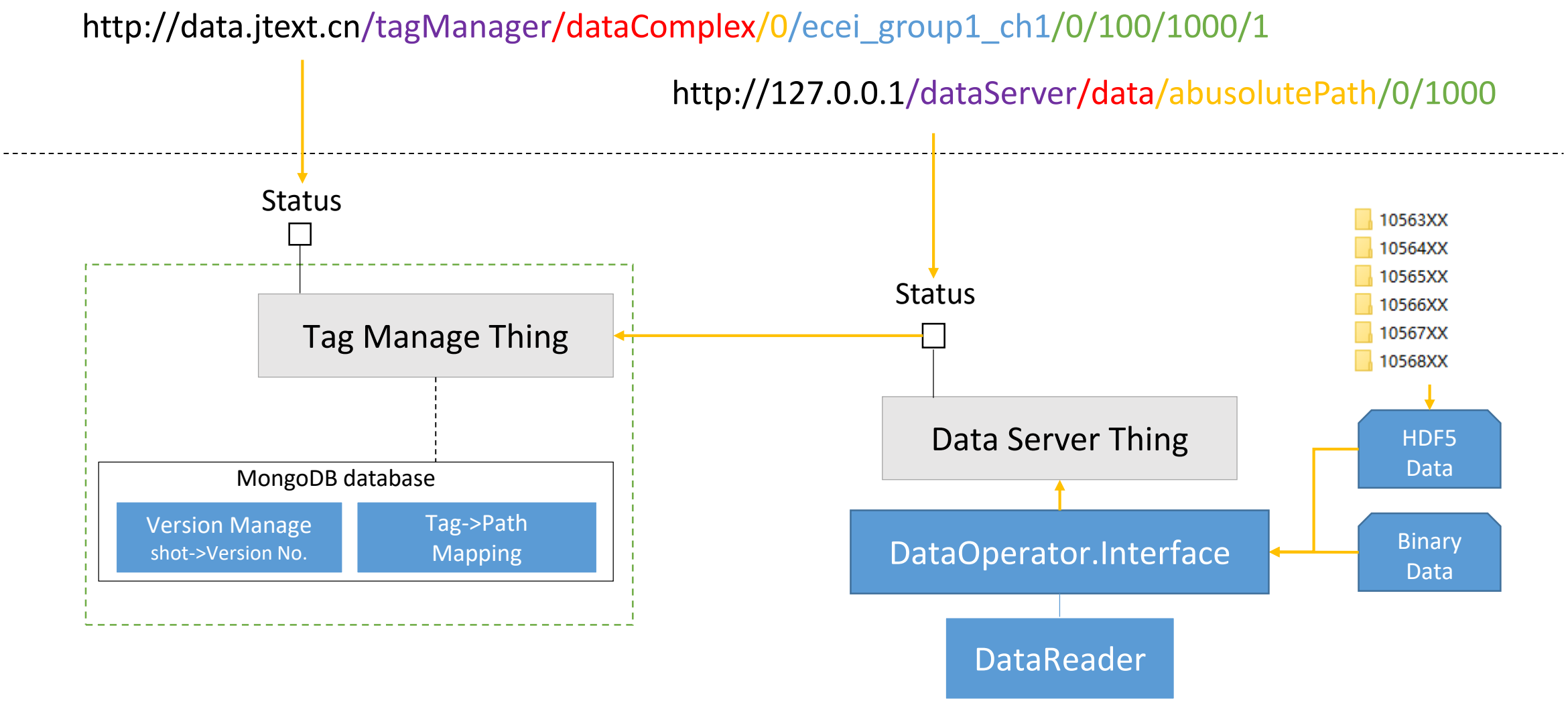
Architecture of DAQ system



Build supports for HDF5 format file



Data archive and access system



Some Status of Tag Manage Thing

Get Create Time of the file: CreateTime(string tag, int shotNo)

Metadata

Get continuous data: Data(string tag, int shotNo, ulong start = 0, ulong length = 0)

Get above's time axis: DataTimeAxis(string tag, int shotNo, ulong start = 0, ulong length = 0)

Get most accurate slice data: DataComplex(

string tag, int shotNo, ulong start, ulong stride, ulong count, ulong block = 1)

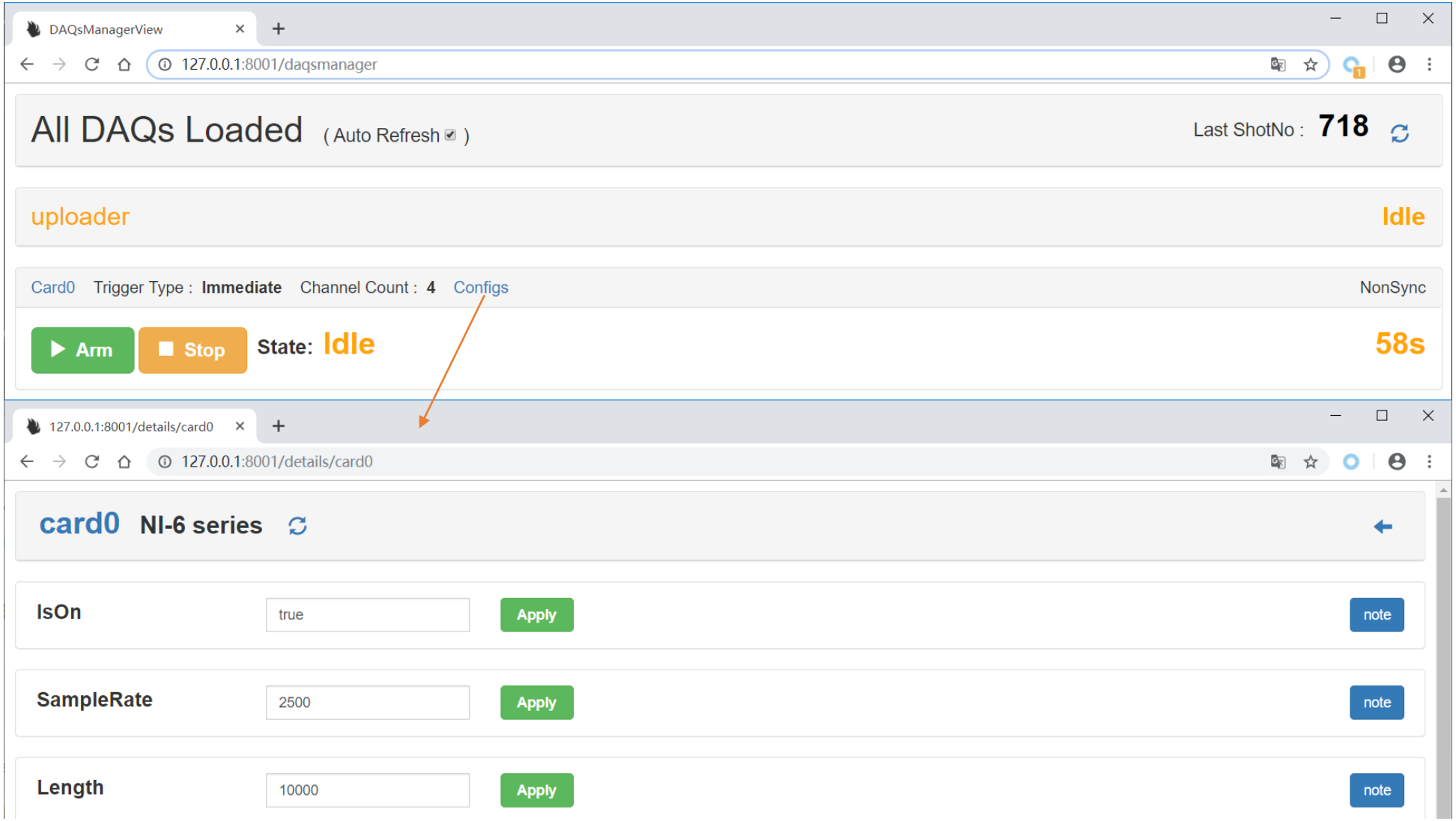
data by sample

Get data by time: DataByTimeFuzzy(

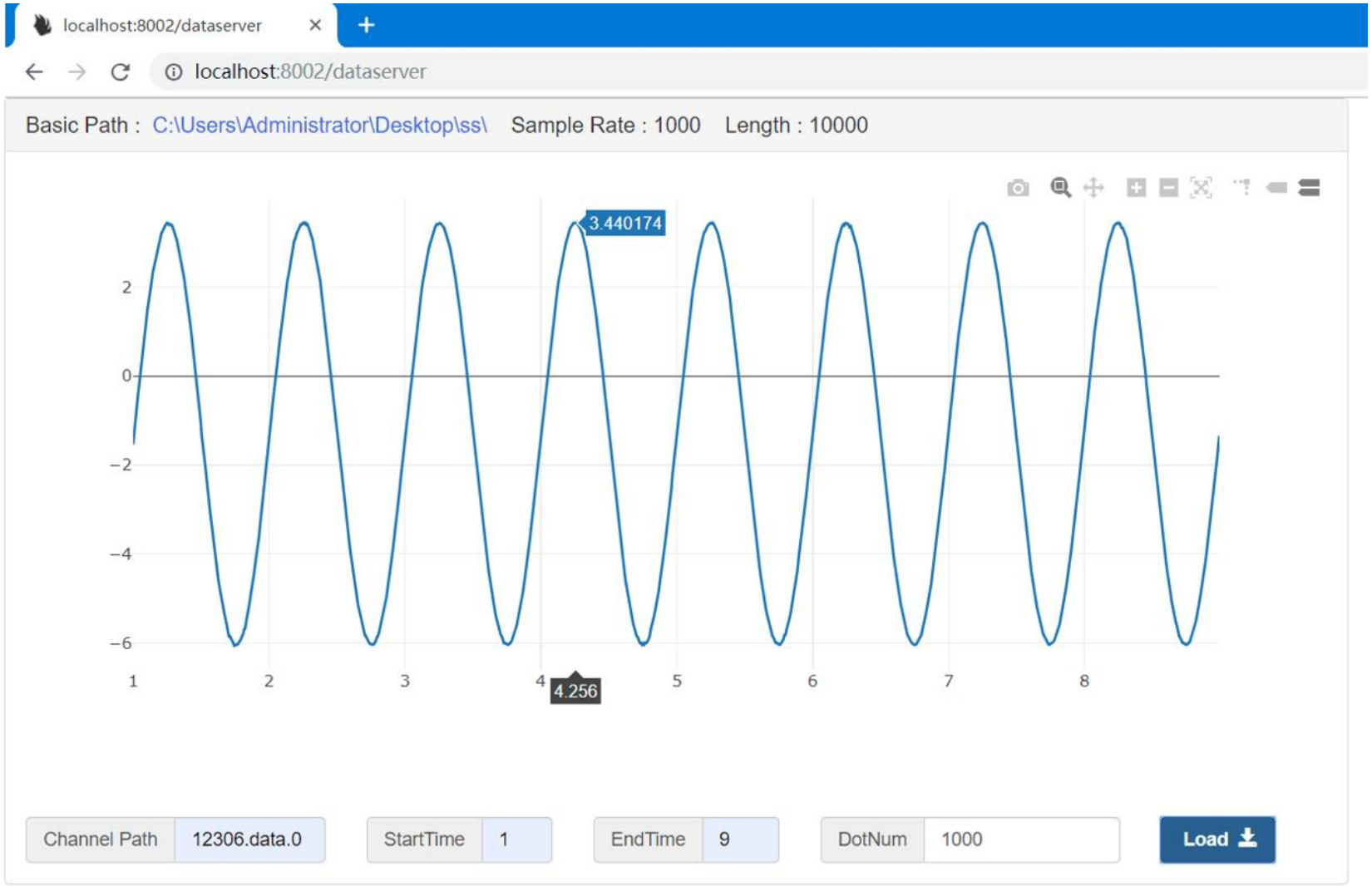
string tag, int shotNo, double startTime, double endTime, ulong count)

data by time

UI of DAQ system in Web



Web Scope for data visualization



Architecture of Web UI

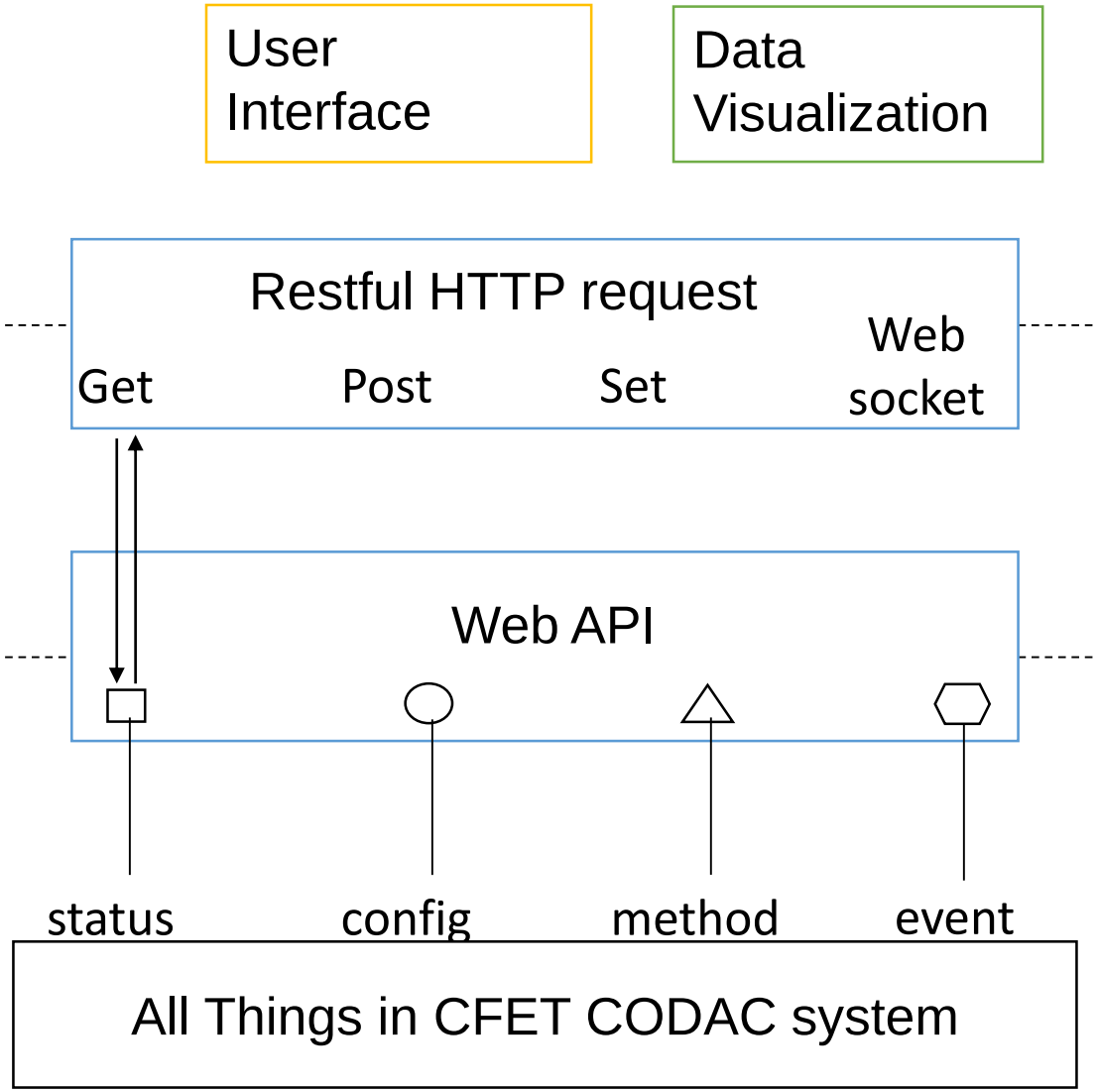
Front-end

- Show information of the CODAC system
- Provide user interface to change state of CODAC system

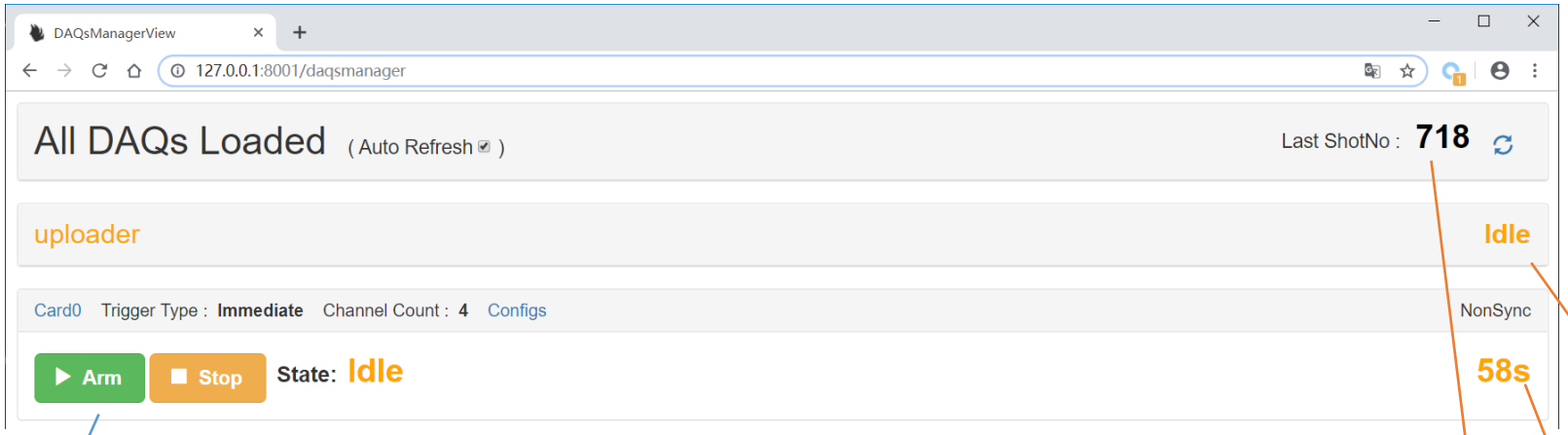
All HTTP requests are **UDA**

Back-end

- Provide CODAC system interfaces
- Implement internal logic



Review UI of DAQ system




```
function arm(index) {
    var s = "" + index;
    $.ajax({
        url: apiArms[s],
        type: "put",
        async: true,
        success: function () {
            //alert(s + ' is armed!')
        }
    });
}
```

Headers Preview Response Timing

General

Request URL: http://127.0.0.1:8001/card0/tryarm

Request Method: PUT

Status Code: 200 OK

```
5 references | wangyuxing, 68 days ago | 1 author, 4 changes
public partial class  AIThing : Thing
{
    /// <summary>
    /// Start AI collecting mission
    /// </summary>
    [Cfet2Method]
    0 references | wangyuxing, 68 days ago | 1 author, 3 changes
    public void TryArm()...
```

```
$.ajax({
    url: apiShot,
    type: "get",
```

```
$.ajax({
    url: apiUpload,
    type: "get",
```

```
$.ajax({
    url: apiTimes[api],
    type: "get",
```

How the Web page is built

- HTML
- CSS
- JavaScript

Bootstrap

jQuery
write less, do more.

Custom design

plotly

Data-Driven Documents

Future work

- **Build support for continuous acquisition**
- **Further testing and application are needed to verify reliability and performance**
- **Web Scope need more further development**
- **Data and metadata in Restful HTTP protocol need to be improved**

Thank you for your attentions

International Joint Research Laboratory of
Magnetic Confinement Fusion and Plasma Physics