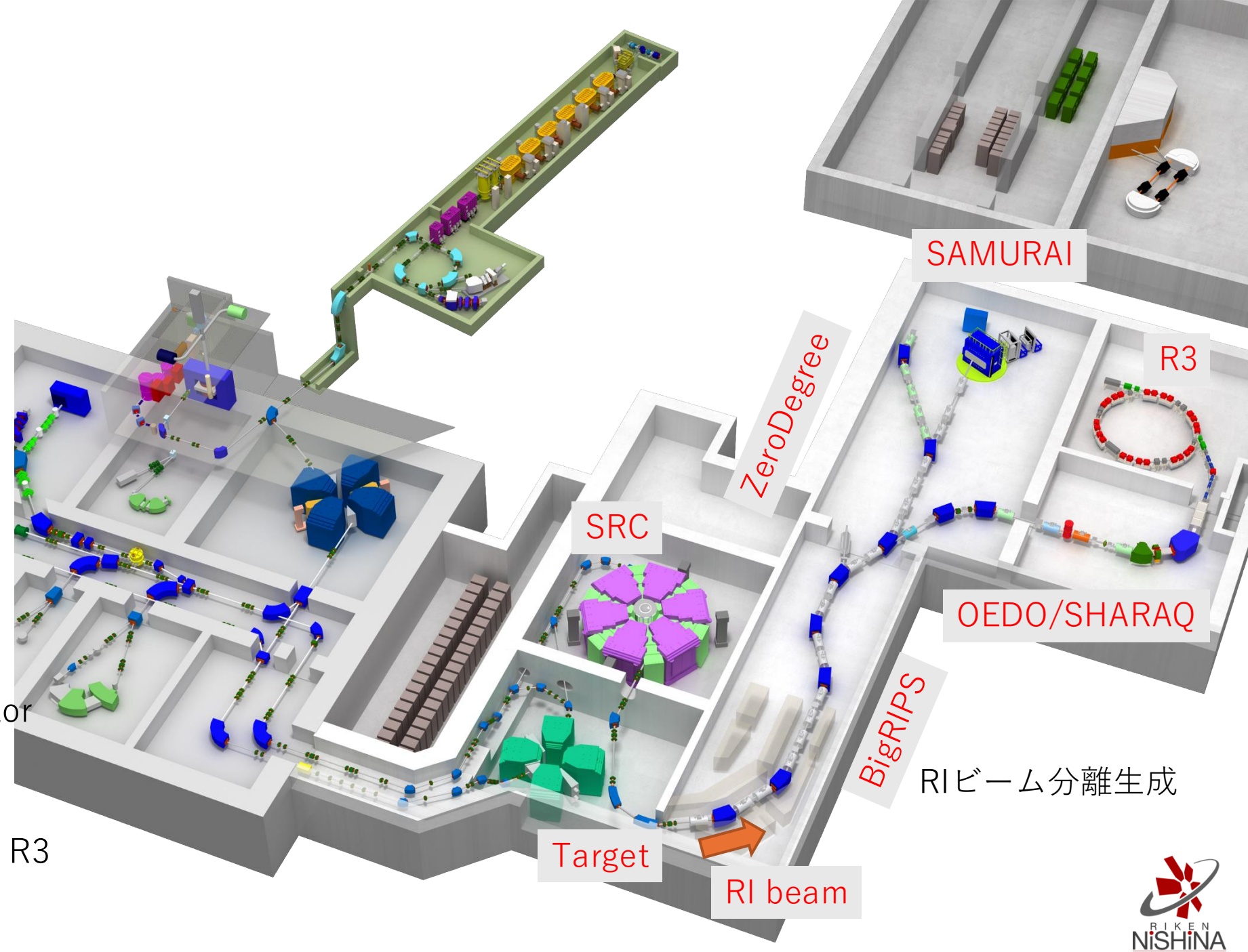
解析・制御の統合プラットフォーム開発による実験遂行能力の高度化

炭竈　聡之
RIKEN Nishina Center

Collaborators: 清水陽平, 馬場秀忠 (RIKEN)

計測システム研究会2025

RI beam @ RIKEN RIBF



In-flight fragment separator
- BigRIPS

Spectrometer etc
- ZeroDegree, SAMURAI, R3
- OEDO/SHARAQ (CNS)

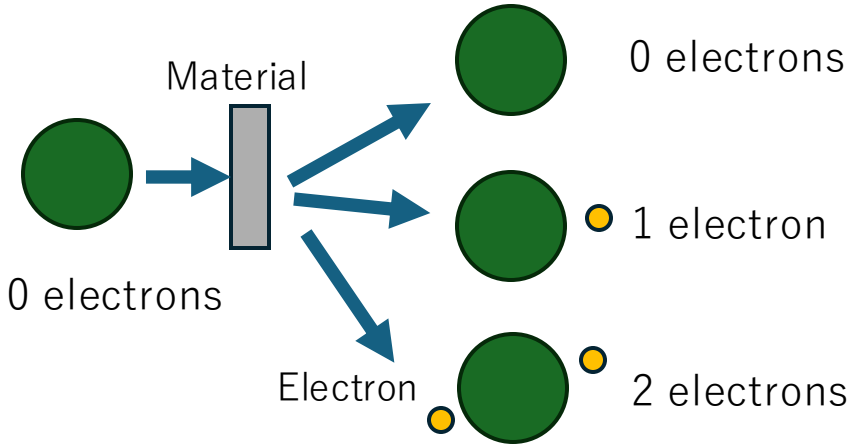
RIビーム分離生成

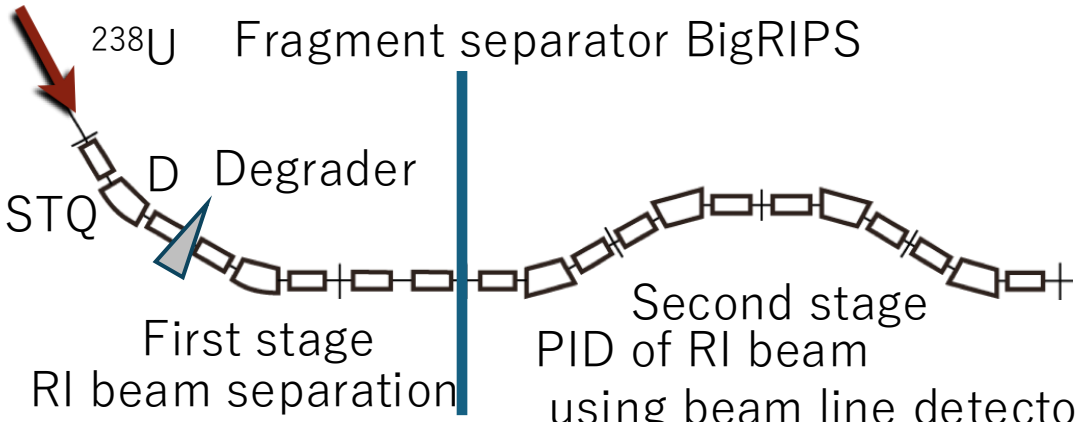
RIKEN NiSHiNA CENTER

Improvement of RI-beam production at RIBF

- Issues about heavy RI beam
  - Charge state changes in beam-line materials.
  - Q change → Bρ change
  - Complicating both the purification and PID of RI beam
- First development of heavy RI beam at RIBF

➢ Heavy-element RI beams





Position: PPAC, Timing: Plastic, Diamond,
Eloss: IC, γ ray: Ge

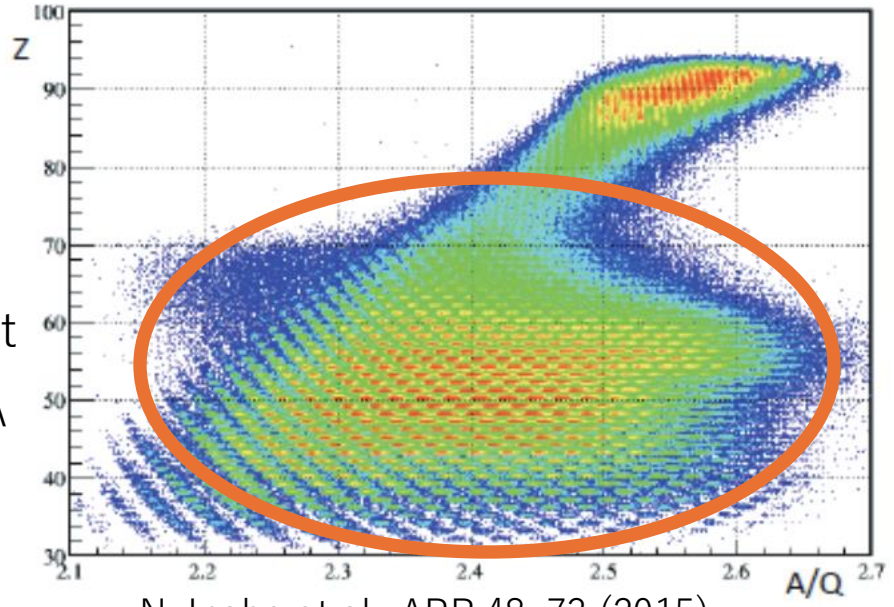
$B\rho - \Delta E - TOF \rightarrow Z, A/Q$

An unexpectedly large background of in-flight fission fragments

Found a practical solution after experiment

100 kHz/pnA

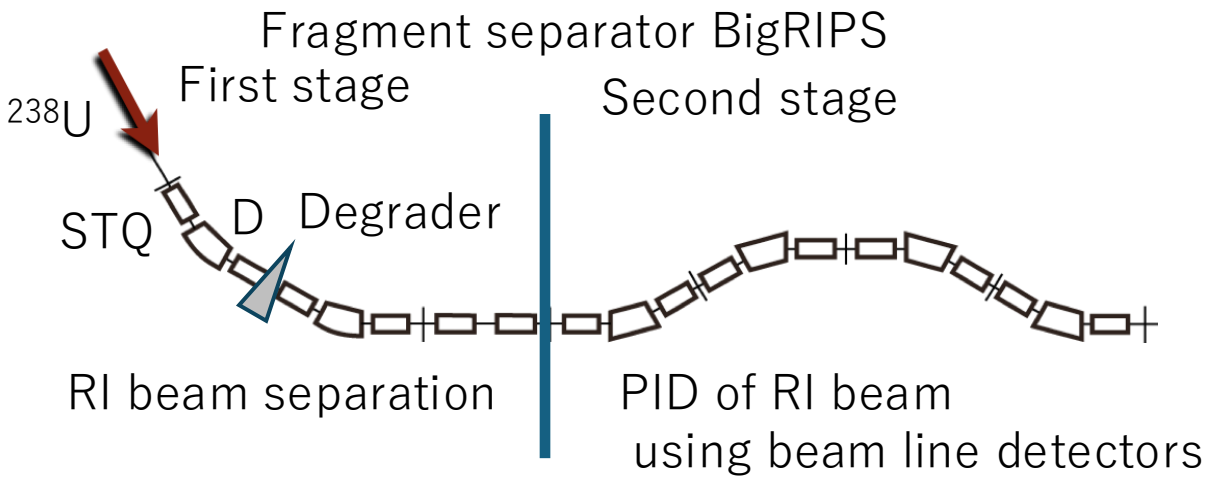
Should be solved online

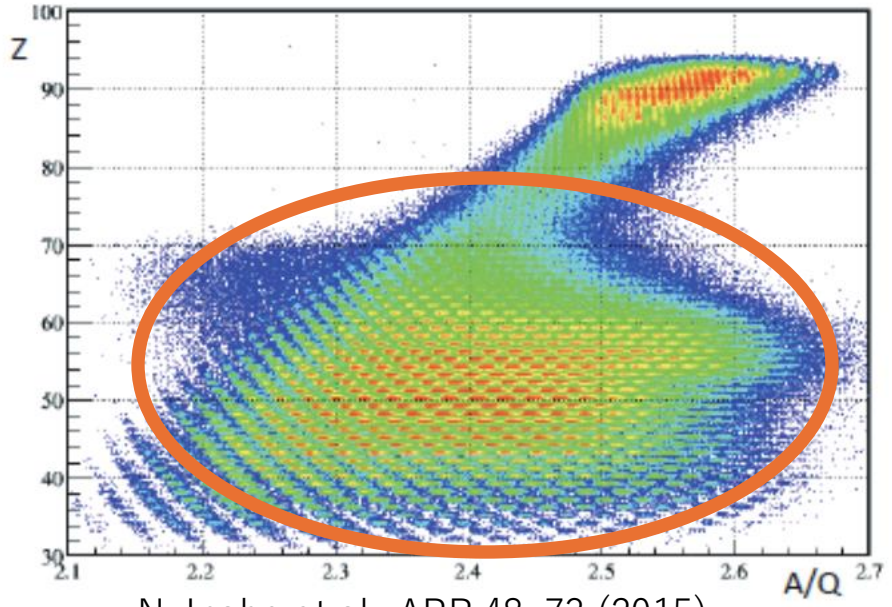


N. Inabe et al., APR 48, 73 (2015)

Improvement of RI-beam production at RIBF

- **Demand for Speed & Quality:** Deliver RI beams *faster* and with *higher quality*.
- **Ideal: Real-Time Optimization:** Fully understand data & adjust BigRIPS.
- **The Reality: Highly Demanding.**
- **Growing Complexity:** More detectors = harder optimization.
- **The Question:** Do we have the right tools?
- **The Goal:** Enhance online capabilities.



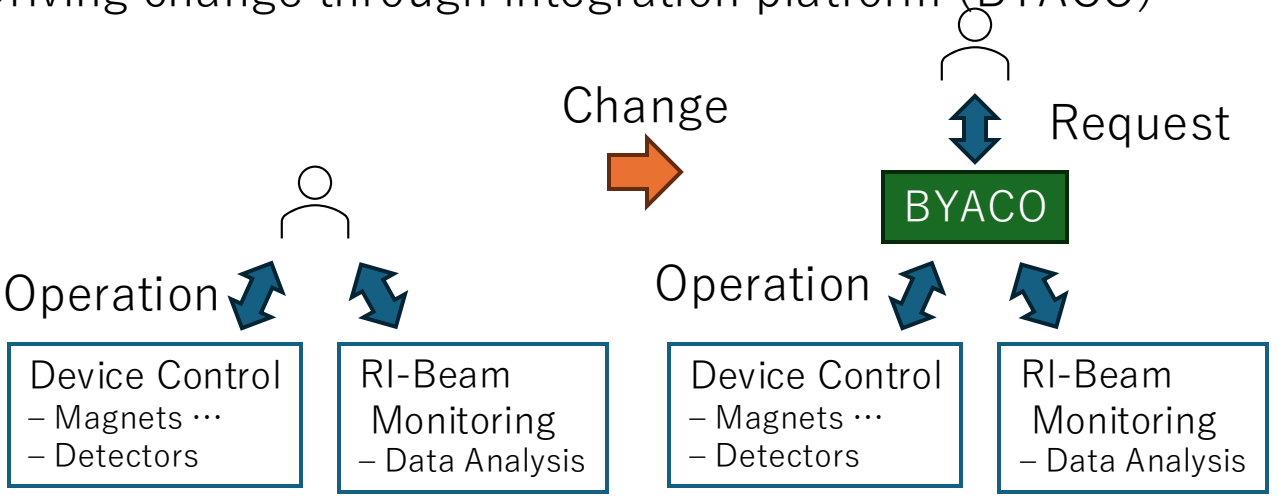


N. Inabe et al., APR 48, 73 (2015)

Vision of integration platform BYACO

- Operate several tasks:
  - Data acquisition
  - Data analysis (histogram, fitting)
  - Magnet currents, detector setting etc.
  - Status monitoring
- Macro
  - Increase productivity
  - Not easy to handle all the matters involved
  - Many macros: Where is it? How to use it?

Driving change through integration platform (BYACO)



T. Sumikama et al., APR 54, 82 (2021).

- BYACO

  (BeYond Analysis, Control or Operation alone)
  - Unifies all relevant controls into a single platform
  - Established workflow
    - Executed by clicking a button in a Web application.

<u>BYACO shifts the human role from routine operation to request.</u>

We can focus on unexpected results and important things during experiment.

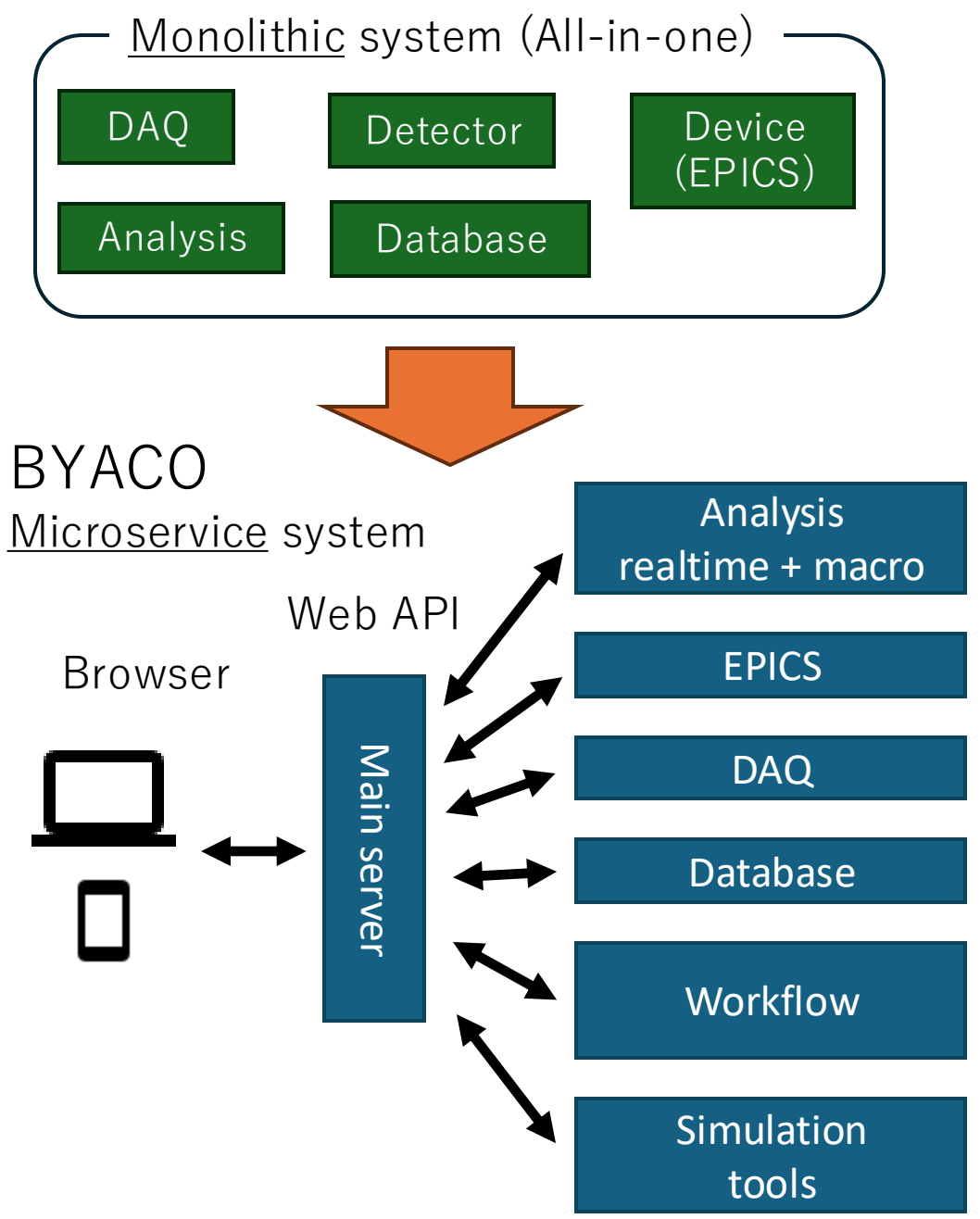
Concept of BYACO

- BYACO: A platform combining multiple functions
  - Designed to advance online experiments, also adaptable to a wide range of use cases.

- Monolithic system
  - As the codebase grows, many components become tightly coupled and making changes slower over time.
  - Locked into a single technology stack.

- Microservice system (BYACO)
  - Loose coupling thorough Web APIs and sockets.
    - Web API (Command)
    - WebSocket (Information sharing)
  - Each component concentrates on a single function
  - Any languages can be used (e.g., C++, Python, JavaScript)
    Note: Some libraries are language-specific.

- Integration of analysis software
  - macros developed for offline analysis



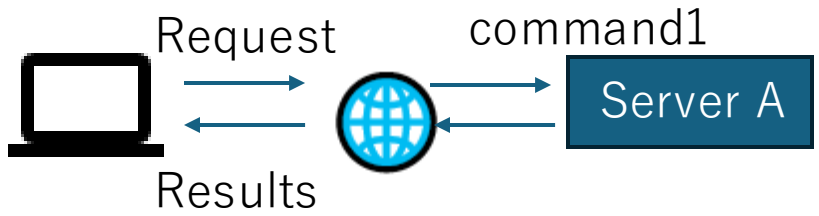
Exposing functionality via Web APIs

1. Use a library for Web server
   - Requires language-specific knowledge

2. Run a standalone program via StdIO
   1. Prepare a standalone program
   2. Modify a Node.js example to expose a Web API
   - This approach is very convenient and our favorite.

Web API

Web API in BYACO
http://byaco.example.internal/serverA/command1

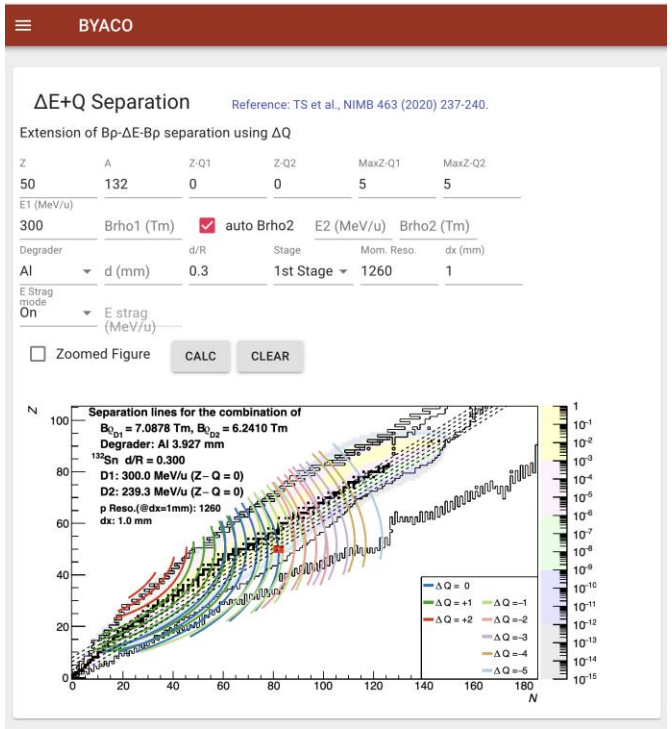


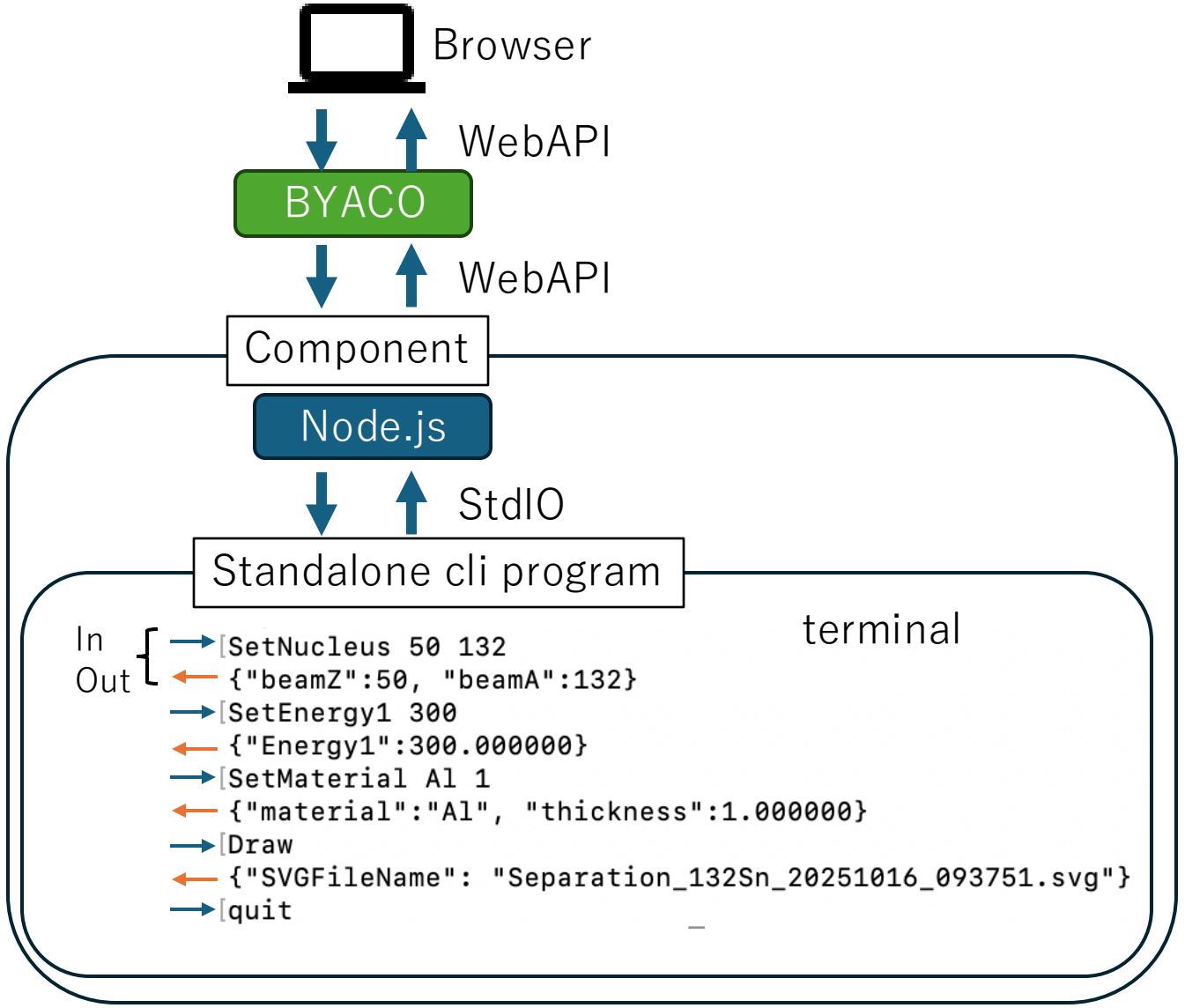
Standard input/output

Fundamental interface
C++: cin/cout
Python: input/print

BYACO Use-Case: Running a simulation & fetching results via Web API

- Simple example
  - Isotope separation of fragment separator in case of Q change



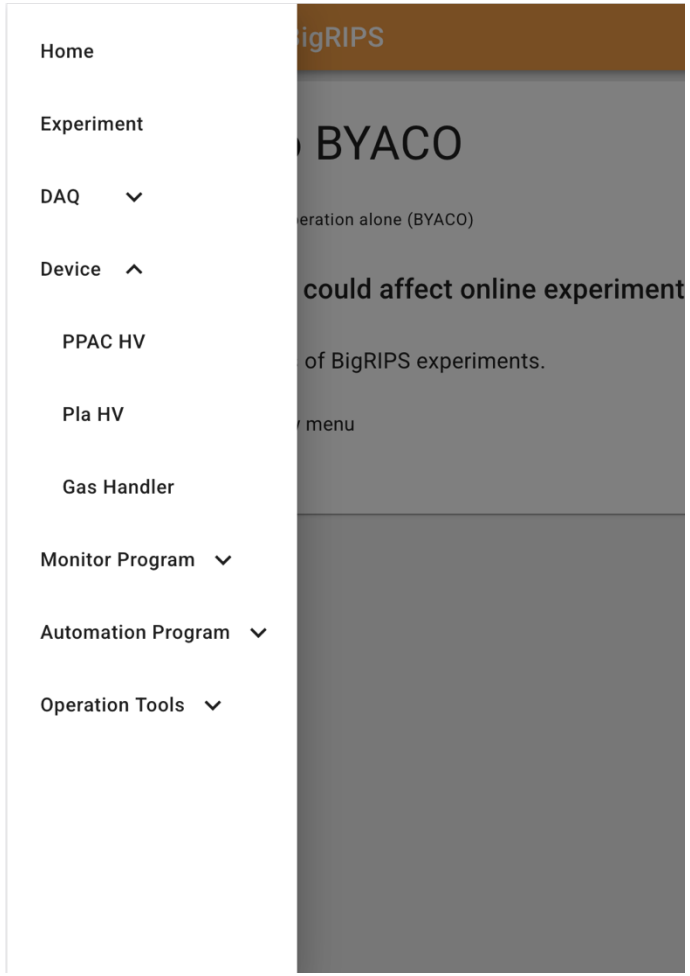


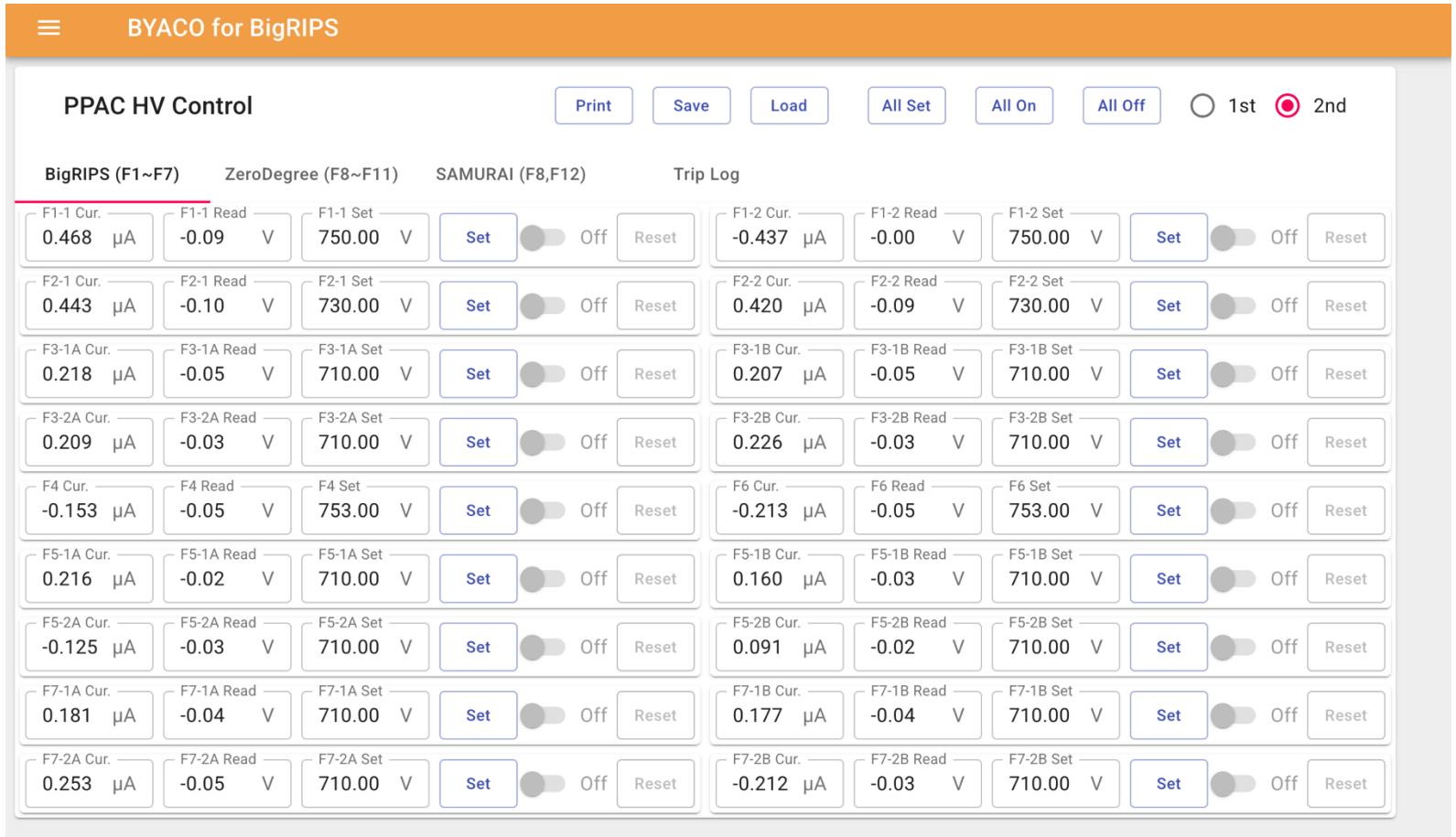
BYACO Use-Cases

Developed by Shimizu-san

Total: about 20 cases

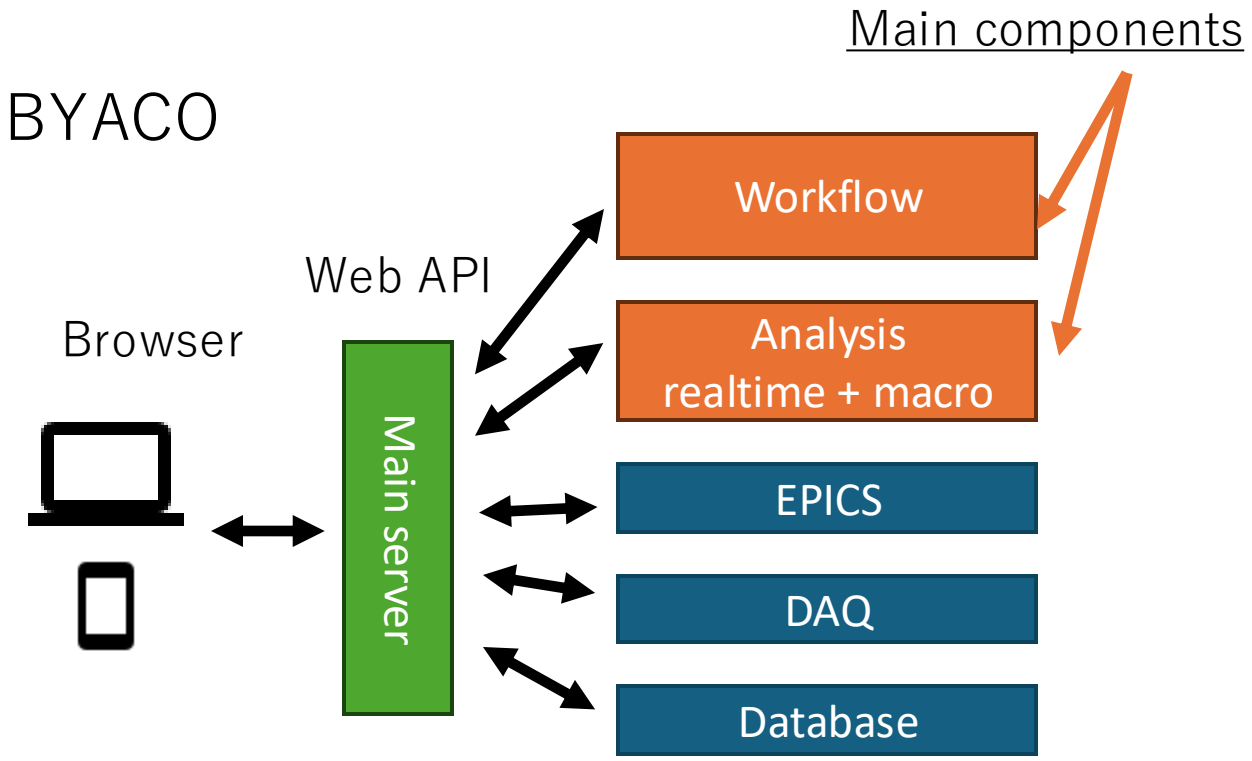
- Detector control





BYACO Use-Case: Automated RI-beam tuning

- Component Design
  - Separate Concerns
    - **Analysis** – applies gates to measured data, perform curve fitting
    - **Workflow Orchestrator** – builds and drives the beam-tuning sequence.
  - Isolate External Interfaces
    - EPICS Interface
    - DAQ Interface
    - Database Interface

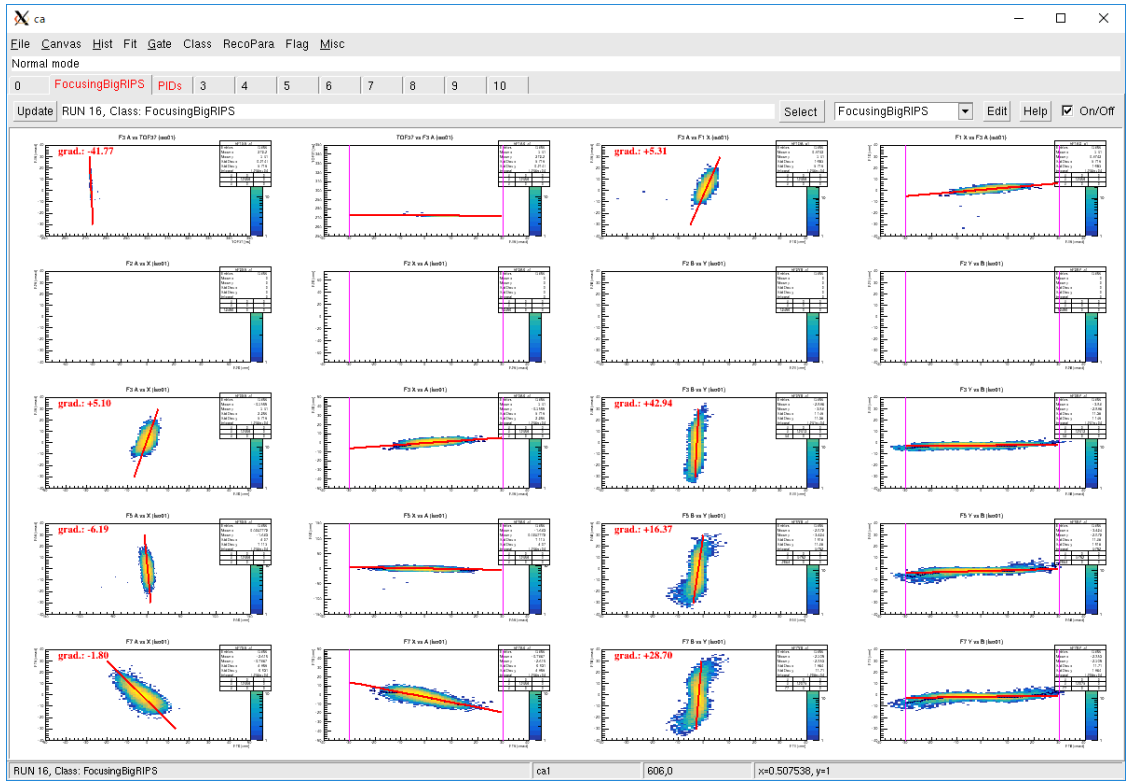


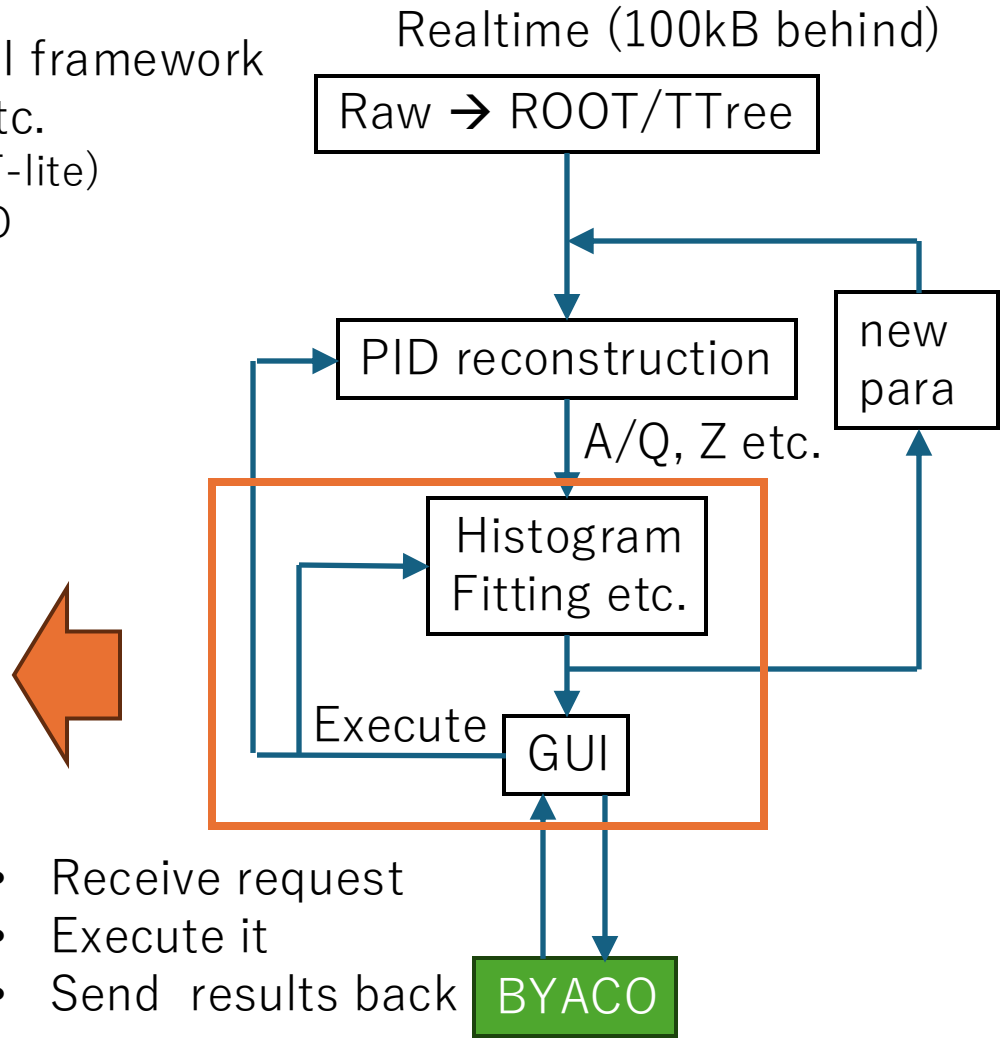
Analysis software with ROOT-based GUI

- Most challenging component
  - Begin with a minimal PoC for automated RI-beam tuning
  - Three sub-components

<u>GUI application</u>

Leverage ROOT's built-in GUI framework
- Histogram filling + fitting etc.
  One tab → One file (PROOF-lite)
- Communication with BYACO





- Receive request
- Execute it
- Send results back

Analysis software with ROOT-based GUI

- Example

Event loop

```
Bool_t FocusingBigRIPS::Process(Long64_t entry)
{
  // The Process() function is called for each entry in the

  if(key.size()==0) return kTRUE;
  fReader.SetLocalEntry(entry);
  if( useflag && !(*flagAll) ){ return kTRUE; }
  if( !IsInside(iso01) ){ return kTRUE; }

  f2x = (*F2X) + f2z * (*F2A) / 1000.;
  f2y = (*F2Y) + f2z * (*F2B) / 1000.;

  f3x = (*F3X) + f3z * (*F3A) / 1000.;
  f3y = (*F3Y) + f3z * (*F3B) / 1000.;

  f5x = (*F5X) + f5z * (*F5A) / 1000.;
  f5y = (*F5Y) + f5z * (*F5B) / 1000.;

  f7x = (*F7X) + f7z * (*F7A) / 1000.;
  f7y = (*F7Y) + f7z * (*F7B) / 1000.;

  // user area
  hF1XA->Fill(*TOF37,*F3A);
  hF1AX->Fill(*F3A,*TOF37);
  hF2XA->Fill(f2x,*F2A);
  hF2YB->Fill(f2y,*F2B);
  hF2AX->Fill(*F2A,f2x);
  hF2BY->Fill(*F2B,f2y);
  hF3XA->Fill(f3x,*F3A);
  hF3YB->Fill(f3y,*F3B);
```

- Programming in C++
  - Merit: Flexible
  - Demerit: Take a lot of effort
    → Use GUI for simple figures and operations
    → ROOT GUI: End of development

After event loop

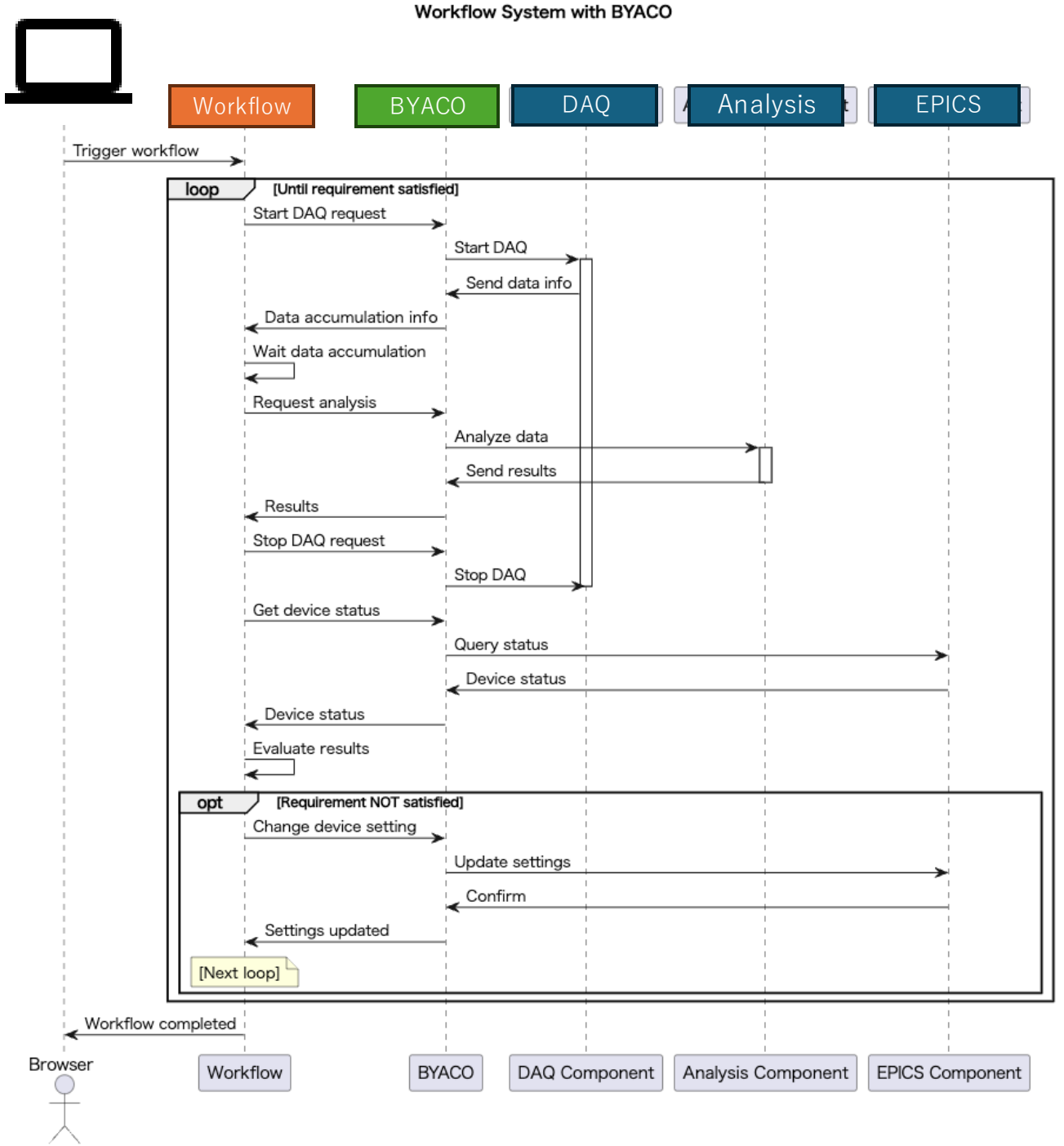
```
void FocusingBigRIPS::Terminate()
  grad->SetTextColor(kRed);

  c->cd(1); hF1XA->Draw("colz"); SetLogz(hF1XA, gPad);
  c->cd(2);
  if( hF1AX->GetEffectiveEntries() > detectionlimit && gf3a.grmin && gf3
    hF1AX->GetXaxis()->SetRangeUser(gf3a.GetMin(),gf3a.GetMax());
    if( hF1AX->GetEffectiveEntries() > detectionlimit ){
      r = hF1AX->Fit("pol1","SQ");
      c->cd(1);
      res->DrawLine(r->Parameter(1)*gf3a.GetMin()+r->Parameter(0),gf3a.G
      grad->DrawLatexNDC(0.15,0.84,Form("grad.: %+.2f", 1/r->Parameter(1
      c->cd(2);
    }
    hF1AX->GetXaxis()->UnZoom();
  }
  hF1AX->Draw("colz"); SetLogz(hF1AX, gPad); gf3a.Draw();
```

Workflow orchestrator

- Workflow communicates with BYACO main server, forwarding requests to each server.

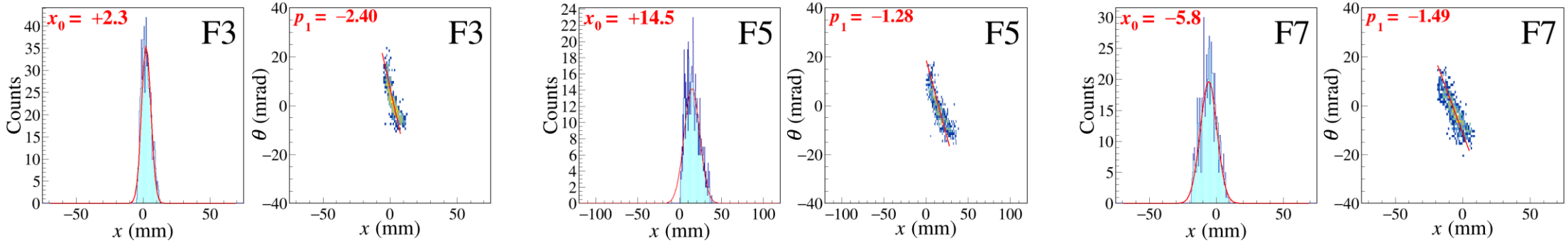
Workflow

- Start DAQ ↔ BYACO
- Wait data accumulation ↔ BYACO
- Request analysis ↔ BYACO
- Wait results ↔ BYACO
- Stop DAQ
  Get information via EPICS
  Evaluate results ↔ BYACO
- Change device via EPICS ↔ BYACO

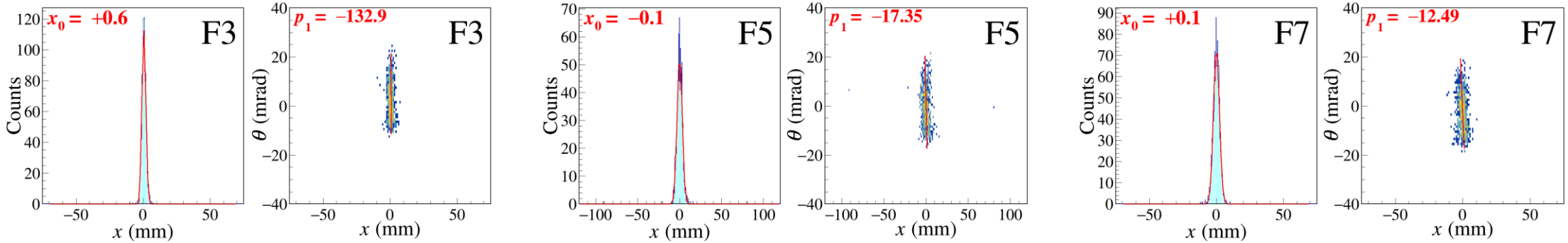


Experimental demonstration
Automated focusing & centering

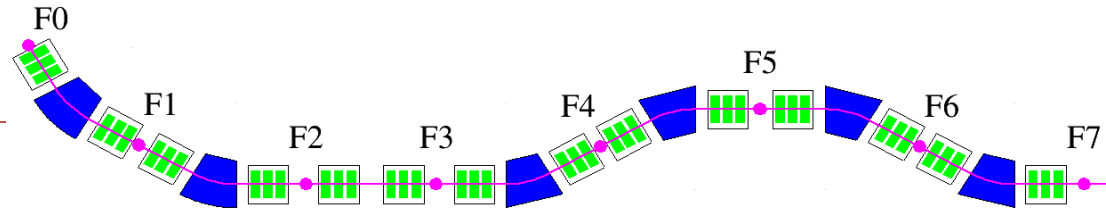
before automated focusing & centering



after automated focusing & centering



Successfully demonstrated automated focus tuning using BYACO.



Courtesy of Y. Shimizu

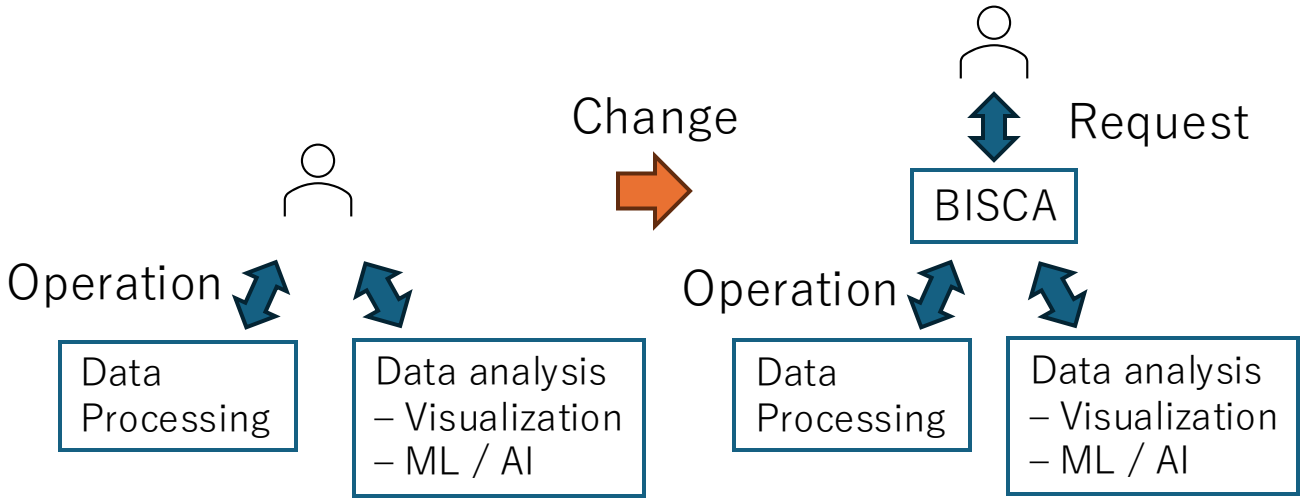
Perspective

- Issues identified in the PoC
  - Analysis:
    RI beams with different Z, A, and E
    Different detector responses
    Different sources of contaminant RI beams

    Before a final RI-beam tuning, we need a detector optimization, rough RI-beam purification and rough RI-beam tuning.
    Analysis software needs to cover these processes and customizations for automation.
    Need flexibility and interactivity.

- Future development

1. New analysis software
   - Bring Inspired Sense and Creativity to Analyst (BISCA)



   - Modern visualization with flexibility and interactivity
   - AI, ML, generative AI

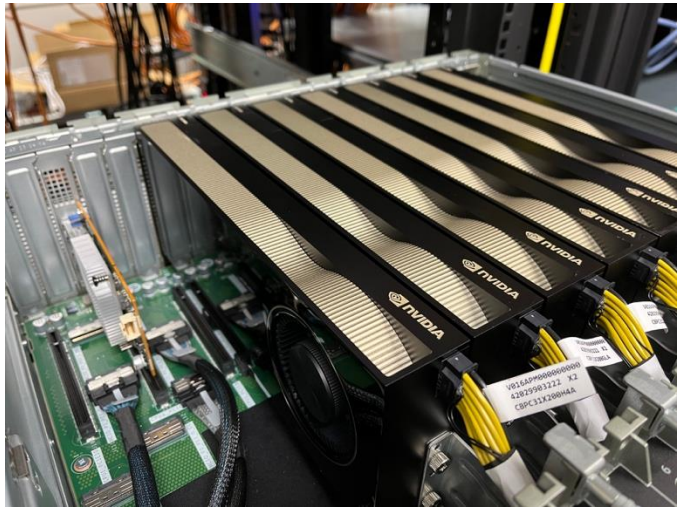
Under conceptual design

Perspective

- Issues identified in the PoC
  - Workflow:
    Use of dedicated OSS
  - Secure Usability:
    Authentication & Authorization
  - Operations:
    Managing an expanding micro-service ecosystem

- Future development

2. Build a Kubernetes cluster for microservices
   - Accelerate migration to a fully containerized stack
   - Enable Kubernetes-native OSS for future integration of AI/ML tools



Summary

- **BYACO is an integration platform** for analysis, control, and operation in RI beam experiments.
- **Microservice architecture** promotes flexibility, scalability, and integration of diverse tools & program languages.
- **Decoupled components & Web API communication** enable automated workflows.
- **Successful demonstration of automated beam tuning** validates the core concepts.
- **Future Development**: Expanding analysis software, dedicated OSS workflow, etc.