

EPICSを用いた中性子散乱装置 制御システムの試み

東京大学物性研究所
附属中性子科学研究施設

小田達郎

oda@issp.u-tokyo.ac.jp

Outline

- 背景：物性研究のための中性子散乱実験

事例紹介

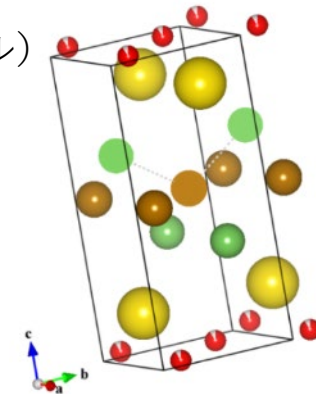
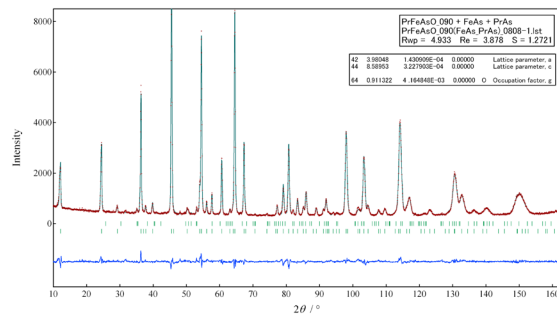
- EPICSによるデバイス制御システム
- Webベース情報モニタ・オンラインデータ解析

中性子散乱分野での事例を紹介し情報共有できれば幸いです

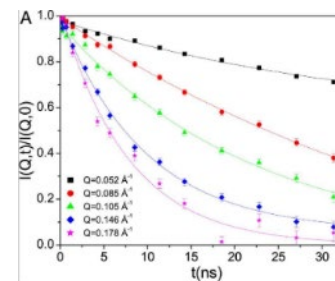
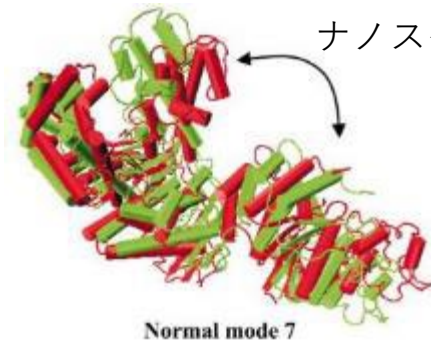
背景：中性子ビームの利用

- 物質構造解析
- 磁性研究
- 物質のダイナミクス研究
- 内部応力測定
- 中性子ラジオグラフィ
- 即発ガンマ線分析
- 医療照射（ホウ素捕捉療法）
- RI製造 など

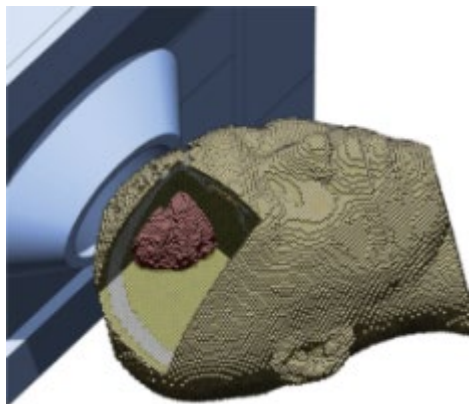
格子定数（Åスケール）



ナノスケール&ナノ秒ダイナミクス

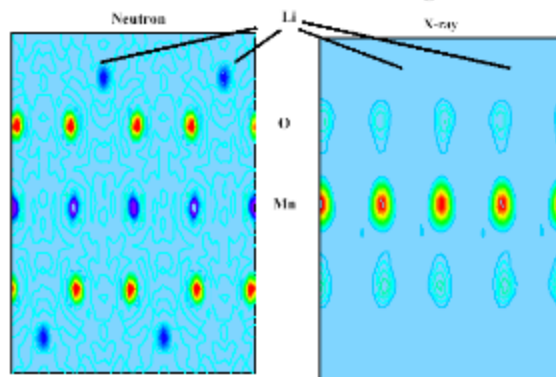


Bu et al., PNAS **102** 17646–17651 (2005)

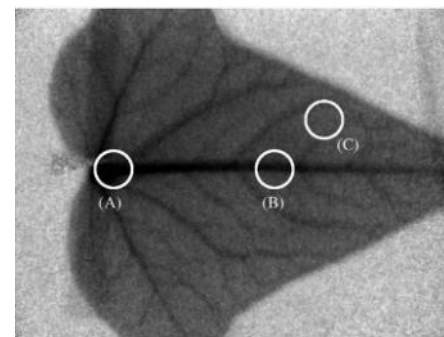


<http://phits.jaea.go.jp/indexj.html>

Li-ion batteries: Neutron is better for detecting Lithium



<https://j-parc.jp/researcher/MatLife/ja/instrumentation/>



U. Matsushima et. al, NIMA 542, 76-80 (2005)

背景：構造・ダイナミクス解析

散乱中性子のエネルギー変化を観測することで特定の空間スケールにおける物質のダイナミクスを知る

Nobel Prize Physics 1994

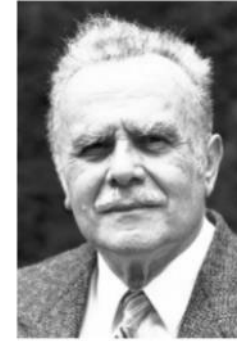
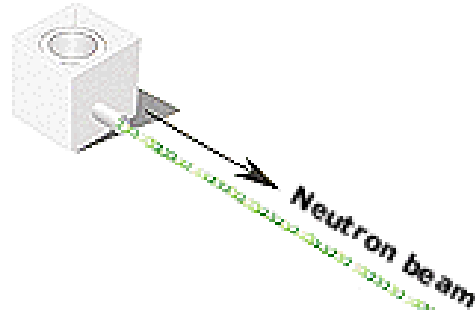


Photo from the Nobel Foundation archive.
Bertram N. Brockhouse
Prize share: 1/2



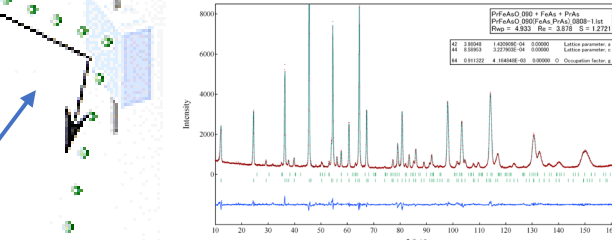
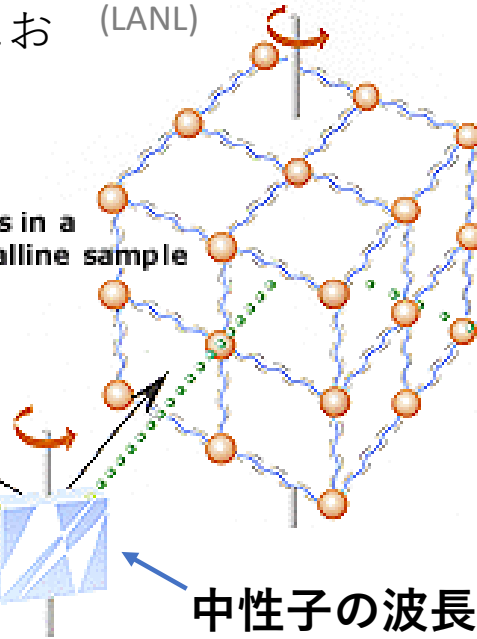
Photo from the Nobel Foundation archive.
Clifford G. Shull
Prize share: 1/2

<https://www.nobelprize.org/prizes/uncategorized/the-nobel-prize-in-physics-1994/>

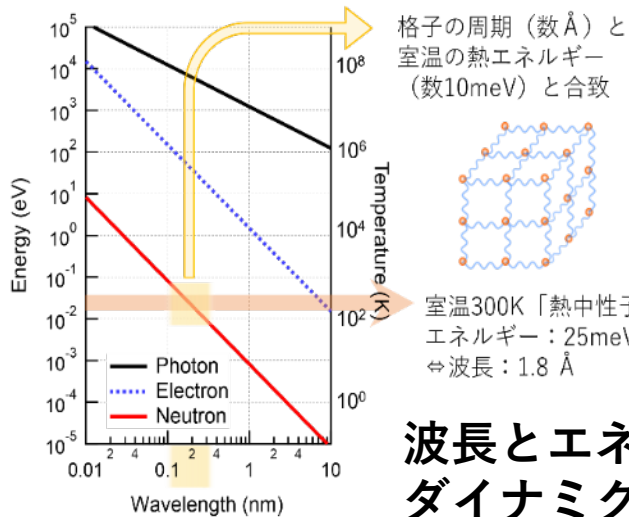


Atoms in a crystalline sample

Ref. Lecture note of Dr. R. Pynn (LANL)

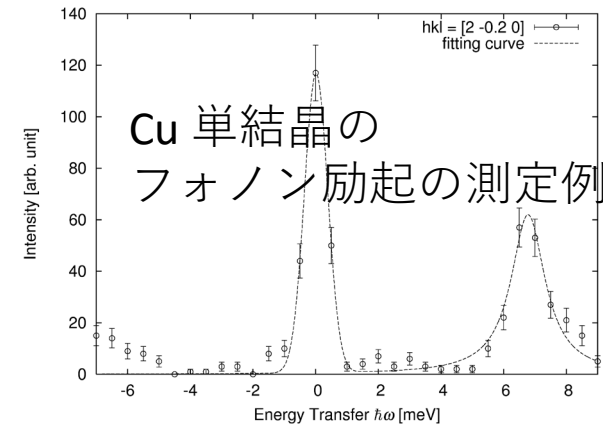


中性子の利点

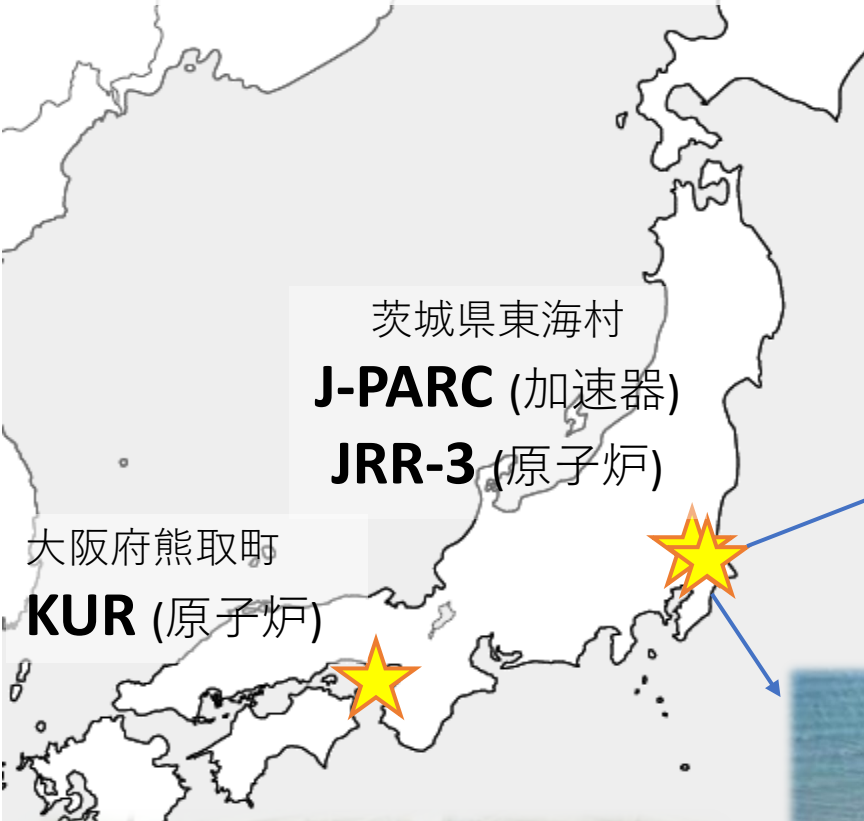


$$\text{Bragg's law: } 2d\sin\theta = n\lambda$$

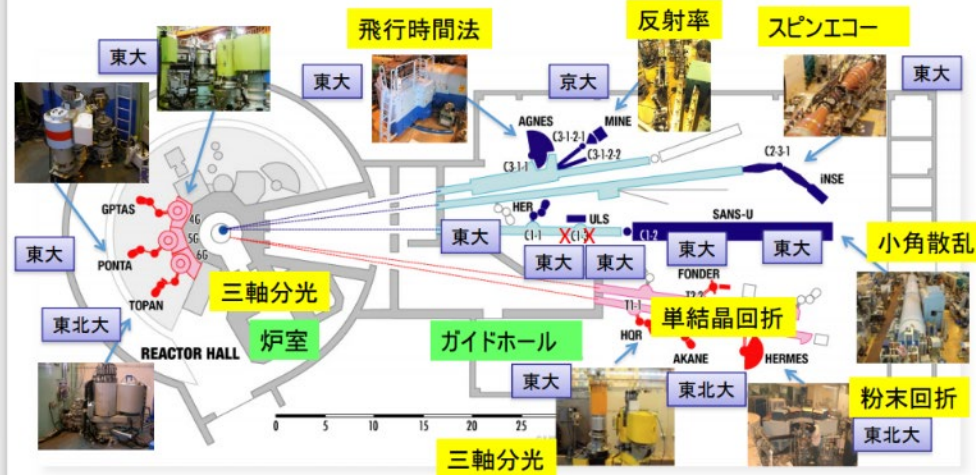
波長とエネルギーの関係がダイナミクス研究に適している



大型中性子実験施設



JRR-3に設置された共同利用装置群



大学所有散乱装置: 12台

東大物性研 10 → 8, 東北大 3, 京大 1

物性研究のほぼ全分野をカバー!

参考: JAEA・QST所有装置
 散乱装置: 14台
 分析装置: 2台
 ラジオグラフィ: 2台

↑中性子科学会将来ビジョン検討委員会の資料から引用

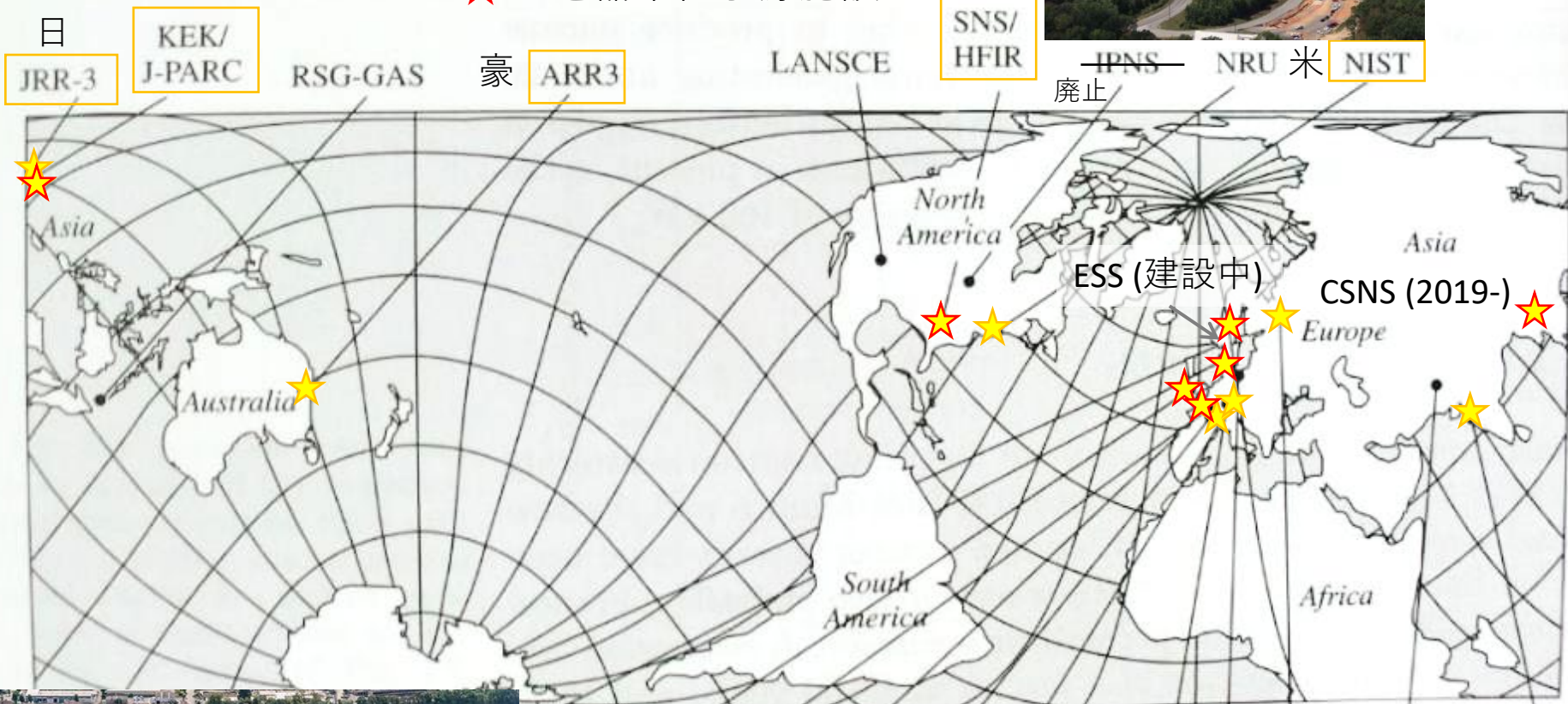


世界の主要な中性子源

★ 加速器中性子源施設



SNSの
写真



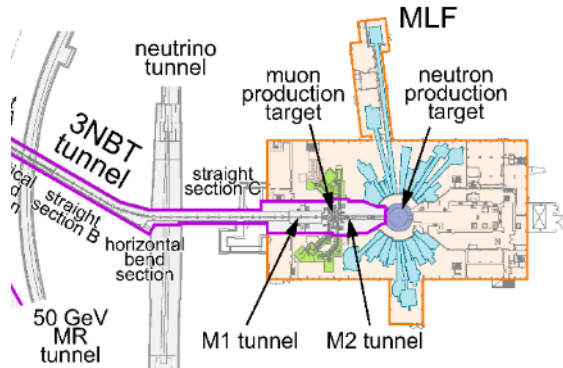
英 ISIS 瑞 SINQ 仏 (廃炉) ORPHEE 独 FRM-II 仏 ILL 独 (廃炉) FRJ-2 露 IBR-2 PINSTECH 印 BARC

世界中からユーザーが
実験しに来る

Ref. "Experimental Neutron Scattering", B. T. M. Willis and C. J. Carlile, Oxford University Press (2013)

仏 ILL の原子炉の写真 (隣は放射光施設 ESRF)

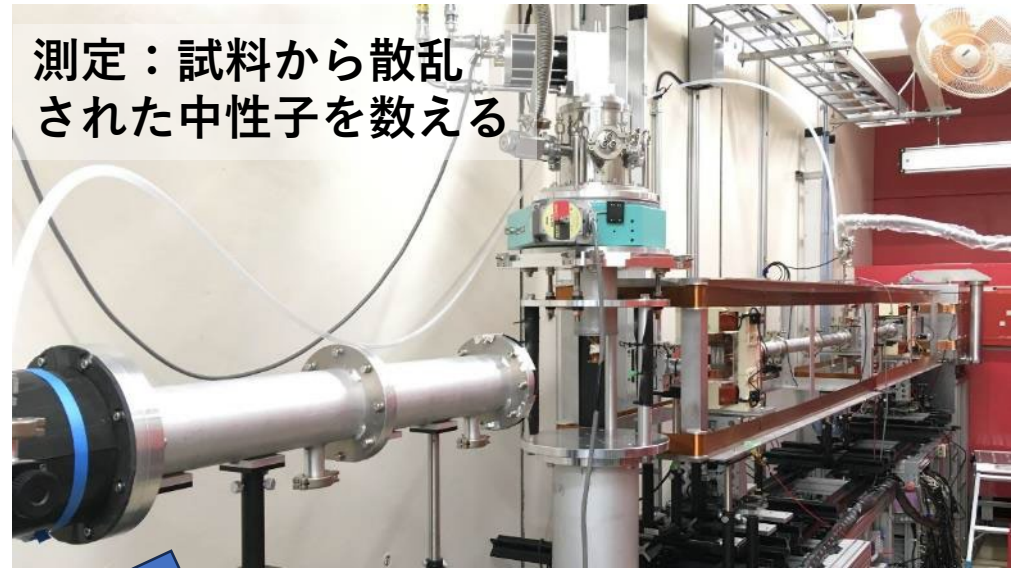
中性子散乱実験のイメージ (J-PARC MLF BL06)



↑ Higemoto et al., Quantum Beam Sci. 2017, 1, 11.



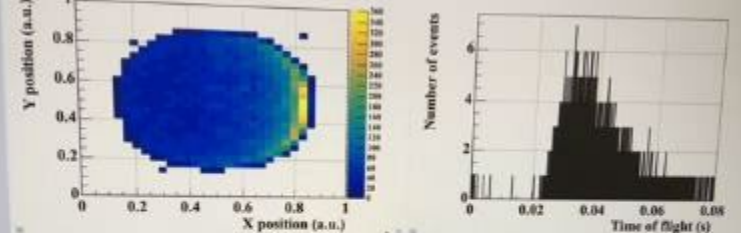
↑ J-PARC MLF BL06 VIN ROSE



データ取得
& リダクション

強度分布

飛行時間スペクトル



強度の経時変化

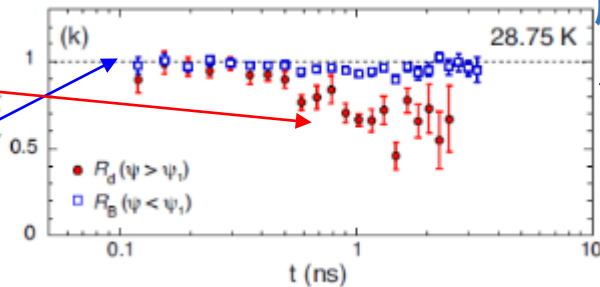
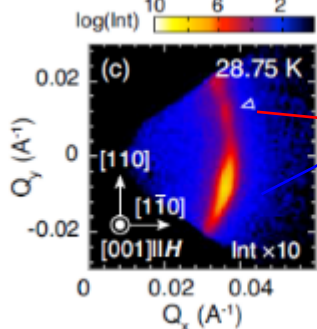
↑のフーリエ変換



解析

逆空間マップ

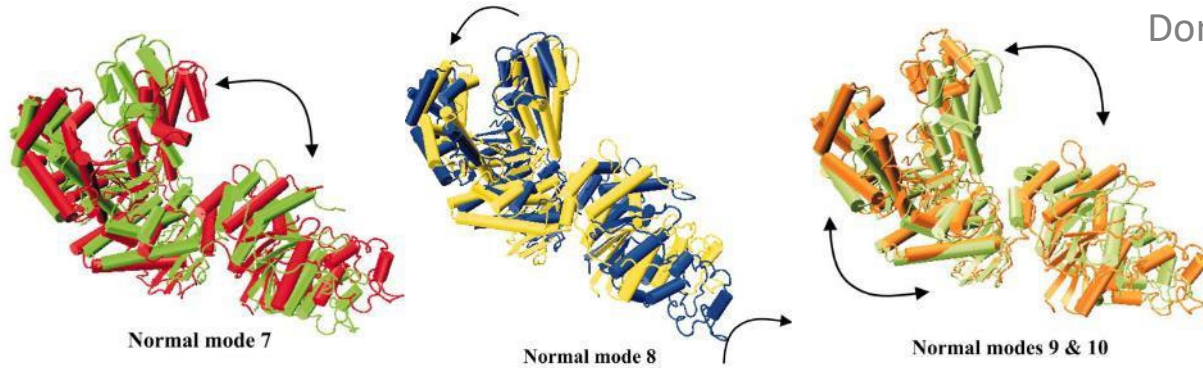
原子位置の時間相関関数
=> 微視的運動状態の情報 => 物性



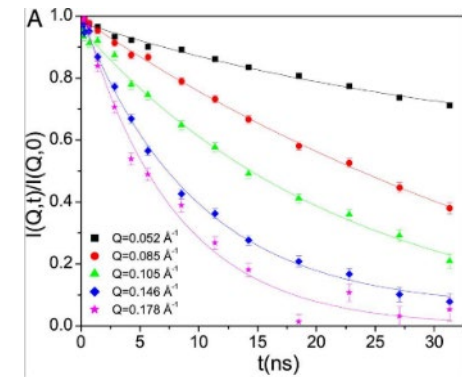
Nakajima et al., Phys. Rev. Research 2 (2020)

中性子スピネコー分光法で見たいものの一例

熱揺らぎの中で高分子はどうして機能を発現できるのか？

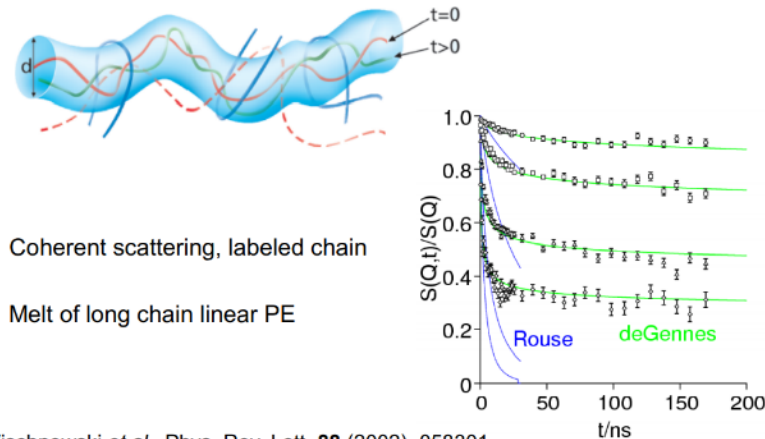


Domain motions of *Taq* polymerase



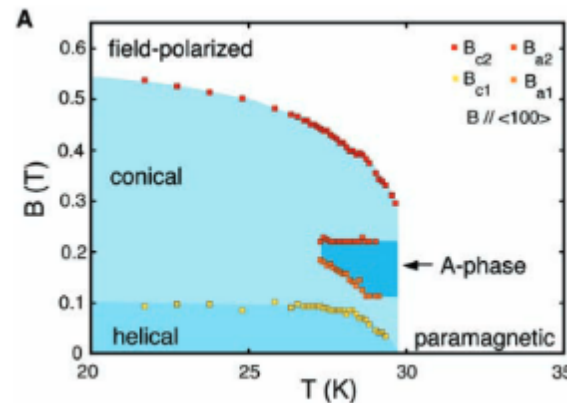
Bu et al., PNAS **102** 17646–17651 (2005)

絡み合った高分子の協同的運動

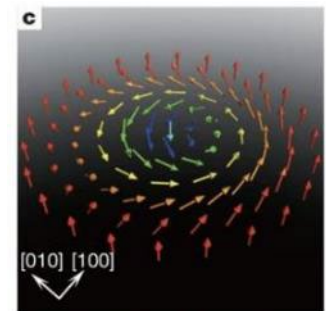


A. Wischnewski et al., Phys. Rev. Lett. **88** (2002), 058301

量子物質の中のスピンドイナミクス



Muhlbauer et al., Science **323**, 915 (2009)

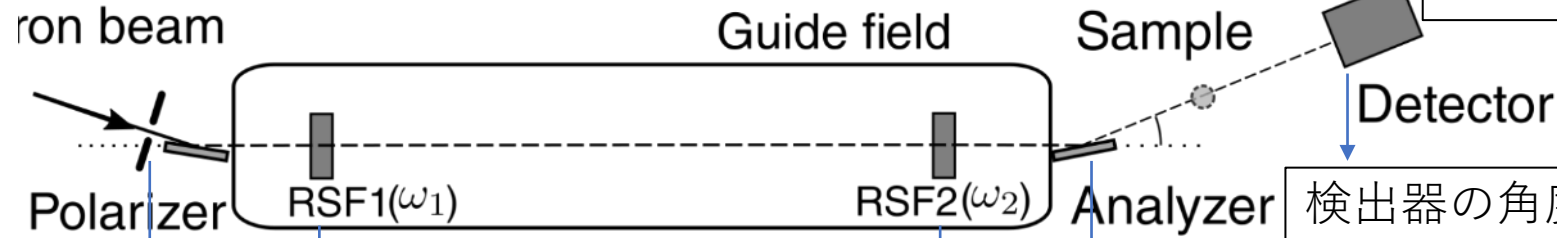


Yu et al., Nature **465**, 901 (2010)

スピンエコー装置調整のイメージ

各デバイスへの命令

キッカーパルス
10000分だけDAQせよ



共鳴スピントリッパー: RSF

「縦磁場コイル」の
「電流」5Aにせよ

「フリッパー1」を
120 mm 上流に動かせ

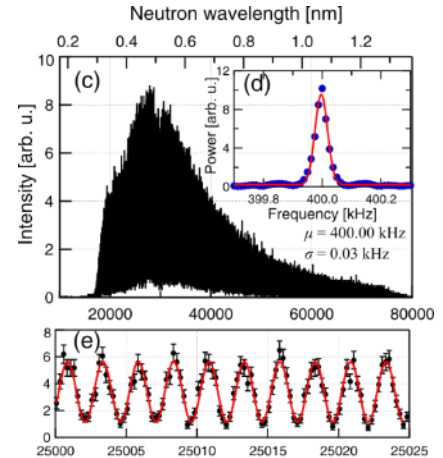
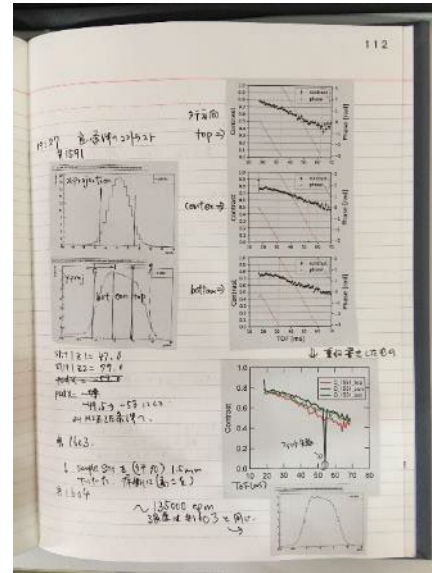
「アナライザミラー」の「角度」を3度に
せよ. X位置を 0.0 mm にせよ

検出器の角度を 1.5度にせ
よ. Y位置を 60mm 後ろに
下げよ

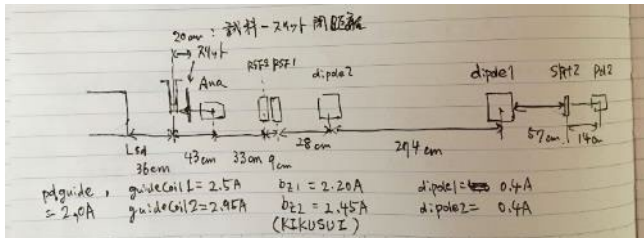
RF 波の Amplitude を 4.0V に
Frequency を 200kHz に
Phase を 0 deg にせよ

「第一スリット幅」を 2 mm にせよ
「X位置」を 0.0 mm にせよ

散乱強度やシグナルの質
をみて実験OKか判断



ログノート例



要求

- モータ，電流，冷凍機，電磁石，中性子検出器などさまざまなデバイスの連動した制御が必要
- 実験毎にユーザ，目的，測定対象（試料）が異なり，使用するデバイス，測定シーケンスも異なる場合がある．フレキシブルな対応が求められる
- 新しいデバイスの追加，それに伴う制御ソフトウェア，ユーザインターフェースの変更を自分たちで行いたい．装置側の人もユーザも使いやすく
- 持続可能性（ライセンス費，引き継ぎの容易さ）



システム開発：KEK物構研の皆様

遠藤仁氏
瀬戸秀紀氏

今回の発表内容は主として安氏の開発による

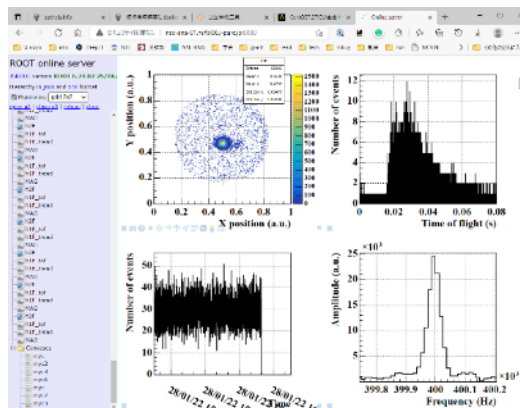
佐藤節夫氏

検出器開発
エレクトロニクス



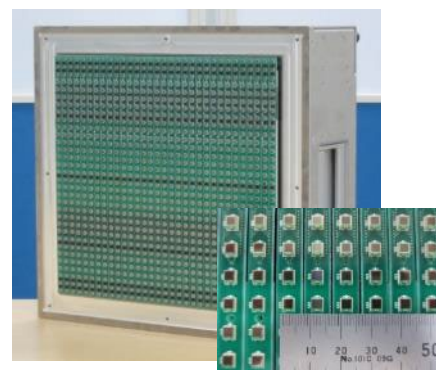
安芳次氏

EPICS制御・DAQ
システム設計・開発



大下英敏氏

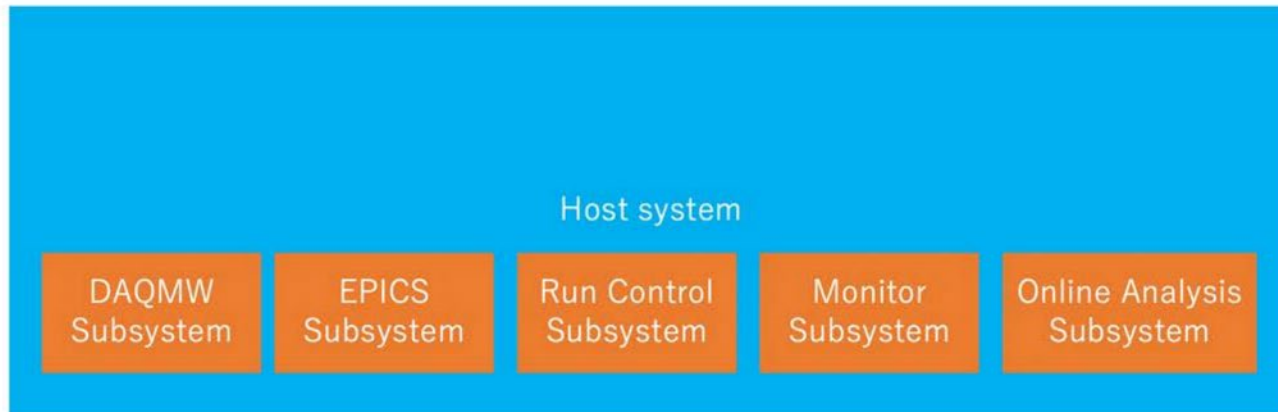
検出器開発
性能評価



瀬谷智洋氏

サーバー
ネットワーク運用

実験制御システムを構成するVirtual machine

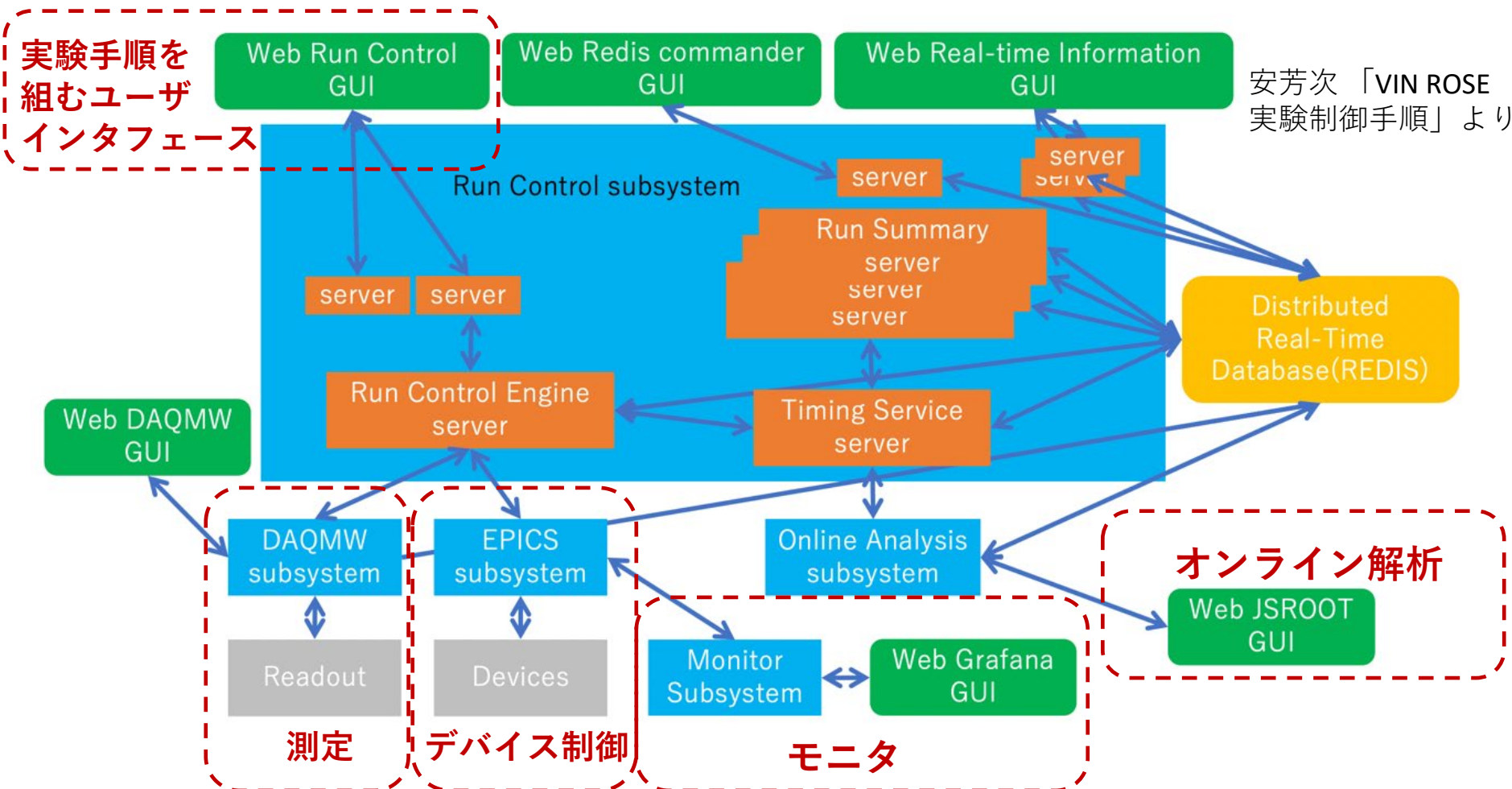


system name	node name	JLAN ip addr	Multilan ip addr	nflslan ip addr	Daqlan ip addr	Devicelan ip addr
Host system	<u>nse-daq-host.j-parc.jp</u>					
DAQMW Subsystem:	<u>nse-daq-00.mlfb106.j-parc.jp</u>					
	<u>nse-daq-01.mlfb106.j-parc.jp</u>					
EPICS Subsystem:	<u>nse-epics-01.mlfb106.j-parc.jp</u>					
Online Analysis Subsystem:	<u>nse-ana-01.mlfb106.j-parc.jp</u>					
Monitor Subsystem:	<u>nse-monitor.mlfb106.j-parc.jp</u>					
Run Control Subsystem:	<u>nse-runcont.mlfb106.j-parc.jp</u>					

なぜ複数VMで構成されるか？

それぞれのサブシステムがそれぞれのソフトウェア依存性を持っていて、すべてをひとつのマシンに詰め込むことや維持することが容易ではない
機能単位にVM化することで、インストールや運用の容易さという点だけでなく、VMをモジュール化してVMの組み合わせによるシステム構成の柔軟性・拡張性が期待できる。複数VMによる複雑さは自動化スクリプトによって隠蔽

制御システムのアーキテクチャ



要求：種々のビーム制御デバイスと中性子検出器のDAQとの連動制御およびオンラインモニタ・データ解析。

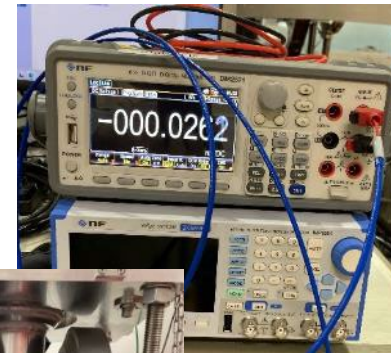
オープンソースソフトウェアによってこれらの機能を実現するための独自ソフトウェア，サブシステム間の連携の仕組みを構築

EPICSで制御しているデバイスの例

- モータコントローラ
（ツジ電子 PM4C, Labo DN2000
など）
- 直流電源（菊水 PWR400など）
- ファンクションジェネレータ
（NF回路 WF1968, WF1984）
- 温度コントローラ
（Lakeshore LS350）
- データロガー
（HIOKI LR8450）
- 温度・湿度計
（T&D おんどとり TR73U）



<https://www.tsuji-denshi.co.jp/product/lineup/stepping/>



冷凍機



デバイスとはEthernetで
通信できるようにしておく

https://www.hioki.co.jp/jp/products/detail/?product_key=1794

PCASpy

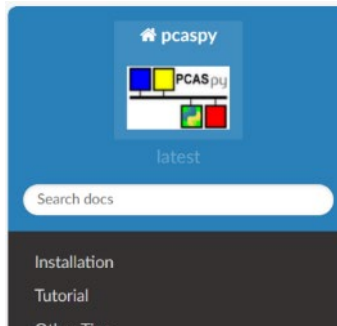
直流電源（Kikusui）の制御の例

Kikusui.py

```
@synchronized
def setCurrent(self, value):
    # CURR
    # value : 0.0 ~ MAXimum(20.0)
    ret = 'ok'
    self.logger.info('CURR '+value)
    status = self.sendCommand('CURR '+value)
    # It makes the output stable in 200msec
    time.sleep(0.2)
    if status != 0:
        ret = 'fail'
    return ret

@synchronized
def getCurrent(self):
    # CURR?
    ret = 'ok'
    self.logger.info('CURR?')
    status, ans = self.sendQuery('CURR?')
    if status != 0:
        ret = 'fail'
    return ret, ans
```

↑pythonでコマンド制御
するメソッドを書いておく



Docs » PCASpy Documentation

[Edit on GitHub](#)

PCASpy Documentation

Overview

PCASpy /'pi:kas,pai/ provides not only the low level python binding to EPICS Portable Channel Access Server but also the necessary high level abstraction to ease the server tool programming.

kikusuiepicsserver.py

```
def makePVDB(logicalname):
    text = ""
    "%s:id" : {
        "type" : "string",
        "value" : "0"
    },
    "%s:info" : {
        "type" : "char",
        "count" : 3000,
        "value" : "0"
    },
    "%s:outputstate" : {
        "type" : "string",
        "value" : "0"
    },
    "%s:voltage" : {
        "type" : "string",
        "value" : "0"
    },
    "%s:remote" : {
        "type" : "string",
        "value" : "0"
    }
    """
    pvdb = "{\n"
    namelist = [logicalname]*9
    pvdb += text % tuple(namelist)
    pvdb += "\n}"
    pvobj = json.loads(pvdb)
    return pvobj
```

(中略)

```
server = SimpleServer()
pvobj = makePVDB(args.logicalname)
logger.debug(str(pvobj))
prefix = ''
server.createPV(prefix, pvobj)
driver = myDriver(devconfig, args.logicalname)

# process CA transactions
while True:
    server.process(0.1)
```

```
from pcaspy import Driver, SimpleServer
from Kikusui import Kikusui

class myDriver(Driver):
    def __init__(self, devconfig, logicalname):
        super(myDriver, self).__init__()
        for kikusui in devconfig["Kikusui"]:
            if logicalname in devconfig["Kikusui"][kikusui]["logicalNames"]:
                self.hostname = devconfig["Kikusui"][kikusui]["controller"][0]
                # don't use self.port because Driver class has the same object.
                self.portnumber = devconfig["Kikusui"][kikusui]["controller"][1]
                break
        if logpath == '':
            self.kiku = Kikusui(self.hostname, self.portnumber)
        else:
            self.kiku = Kikusui(self.hostname, self.portnumber, logpath)
        self.logpath = logpath
        self.kiku.connect()

    self.methodTable = {
        'idin':self.idin,
        'infoin':self.infoin,
        'outputstatein':self.outputstatein,
    }
```

```
server = SimpleServer()
pvobj = makePVDB(args.logicalname)
logger.debug(str(pvobj))
prefix = ''
server.createPV(prefix, pvobj)
driver = myDriver(devconfig, args.logicalname)

# process CA transactions
while True:
    server.process(0.1)
```

デバイス設定ファイルの例 dev-config.json

```
"PMCTcp": {   モータコントローラ
  "pmc1": {
    "controller":["192.168.10.xxx", 7777],
    "logicalNames": {
      "axisX":"0",
      "axisY":"1",
      "axisZ":"2",
      "gonio":"3"}
  },
  "Kikusui": {   直流電流
    "kikusui0": {
      "controller":["192.168.10.xxx",5025],
      "logicalNames": {
        "bz1":"0"}
    },
    .....
```

直動ステージ
...

ゴニオメータ

縦磁場

モータのパルス値, 原点,
バックラッシュなどの設定は
別に行う
(例)

```
{
  "initTable": {
    "pmc1": {
      "0": {
        "pulsePerUnit": 250,
        "EngineeringUnit": "mm",
        "pulseForZero": 1000
      },
      ...
    }
  }
}
```


測定シーケンスを組むためのWeb UI (Run Control)

Operation Panel for DAQ operator on 192.168.100.75

Current Status: OK

Run Number: 50

DAQ button:

DAQ status:

group	State	Status	Total Byte Counts
group0.PublisherMIO	LOADED	WORKING	0
group0.LoggerMIO	LOADED	WORKING	0
group0.DispatcherMIO	LOADED	WORKING	0
group0.PlayerEmak0	LOADED	WORKING	0

XML response from DaqOperator

Control Panel of Run Control for VIN ROSE - Mozilla Firefox

Control Panel of R x operatorPanel x Normal Real Time x Special Real Time x Redis Command x Control Panel of R x bl06tr73u - Gr x Online server x +

192.168.100.65:8000/file?filename=sample1.json

Control Panel of Run Control for VIN ROSE

System Status

- Run Control Subsystem
- Timing Service Subsystem
- EPICS Subsystem
- DAQMW Subsystem

Run Mode

Dry Run
 Step by Step

Run Start/Stop

Select Run Script

sample1.json

Load Panel

Load: ok

```
1 |
2 | {"comment":"MpixX:0.0,40.0 intervals:4 kp:10000 runNo:50,54"},
3 | {"var":{"$result":0}},
4 | {"redis":{"set":["RunInfo:SampleId","09040006241"]}},
5 | {"redis":{"set":["RunInfo:SubstanceName","Praseodymium-Nikel(PrNi2)"]}},
6 | {"epics":{"caput":["MpixX:position","0.0"]}},
7 | {"epics":{"caget":["MpixX:position","$result"]}},
8 | {"daq":{"kp":10000,"runNo":50}},
9 | {"epics":{"caput":["MpixX:position","10.0"]}},
10 | {"epics":{"caget":["MpixX:position","$result"]}},
11 | {"daq":{"kp":10000,"runNo":51}},
12 | {"epics":{"caput":["MpixX:position","20.0"]}},
13 | {"epics":{"caget":["MpixX:position","$result"]}},
14 | {"daq":{"kp":10000,"runNo":52}},
15 | {"epics":{"caput":["MpixX:position","30.0"]}},
16 | {"epics":{"caget":["MpixX:position","$result"]}},
17 | {"daq":{"kp":10000,"runNo":53}},
18 | {"epics":{"caput":["MpixX:position","40.0"]}},
19 | {"epics":{"caget":["MpixX:position","$result"]}},
20 | {"daq":{"kp":10000,"runNo":54}}
21 |
22 |
```

variable,
loop, double loop,
Shell command,
etc...

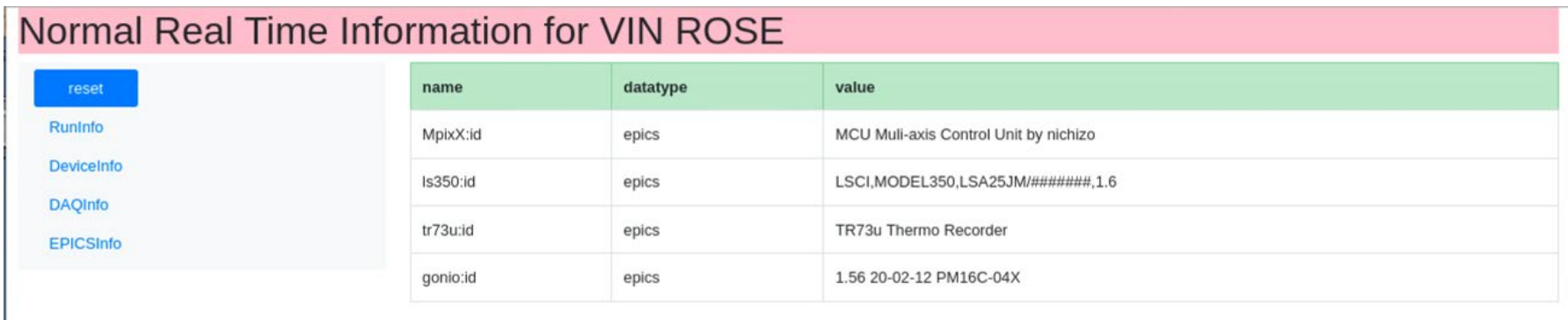
```
[daq@nse-runcont ~]$ tail -f /work/logs/runcont/runcontrolengine.log
2021-11-30T11:45:40.547899+09:00 [Daq_Run] info DAQMW_setRunNumber is OK
2021-11-30T11:45:40.606579+09:00 [Daq_Run] info DAQMW_Begin command is OK
2021-11-30T11:45:40.623388+09:00 [Daq_Run] info DAQMW_RUNNING state is confirmed
2021-11-30T11:47:22.064306+09:00 [Daq_Run] info DAQMW_Stop command is OK
2021-11-30T11:47:22.087830+09:00 [Daq_Run] info DAQMW_CONFIGURED state is confirmed
2021-11-30T11:47:22.087956+09:00 [Run Control Subsystem] info daqrun:done
2021-11-30T11:47:22.100581+09:00 [Run Control Subsystem] info postprocess:sucess
2021-11-30T11:47:32.125673+09:00 [Run Control Subsystem] info postprocess:watchsubscribersdone:done
2021-11-30T11:47:32.127147+09:00 [Run Control Subsystem] info run_end
2021-11-30T11:47:32.127606+09:00 [Run Control Subsystem] info wait for runscript command
```

shellinabox

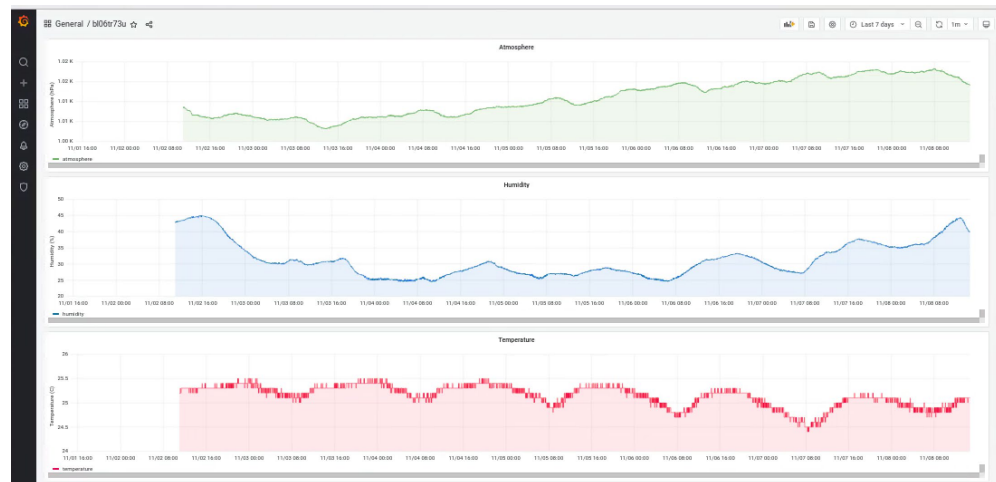
安芳次「VIN ROSE 実験制御手順」より

実験情報モニタ

- EPICS ioc と redis の中にあるリアルタイム情報を取得し表示する仕掛け (based on python redis module)



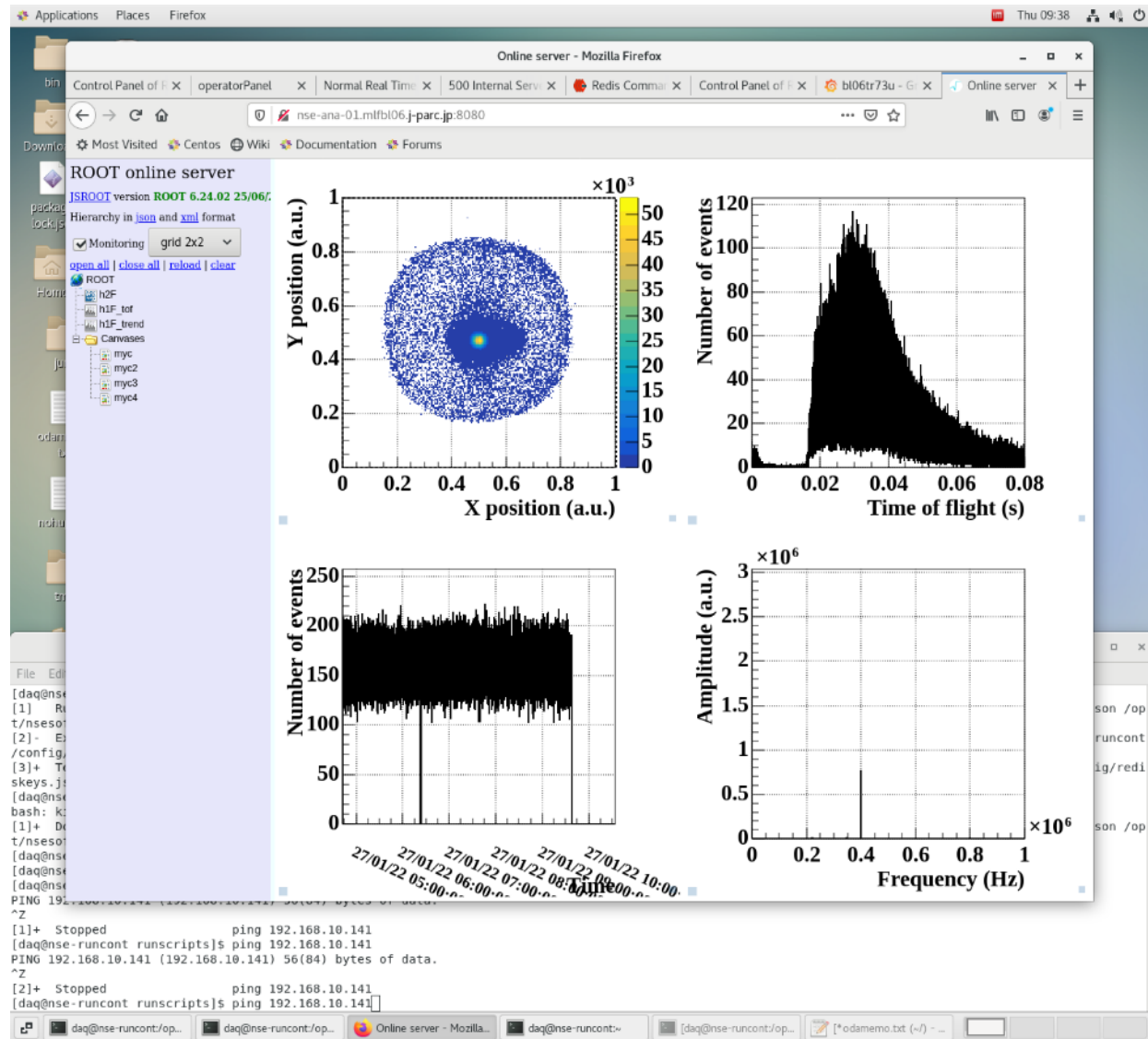
- EPICSサーバが取得した情報をmysqlに書き込んでGrafanaで表示



オンラインデータ解析 by JS ROOT

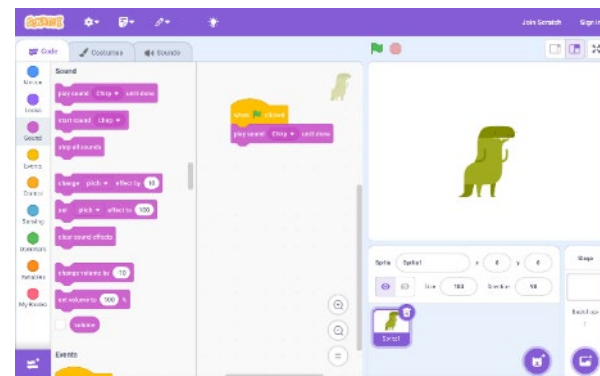
- 単純な2次元マップ
- 飛行時間ヒストグラム
- スピンエコーシグナルをフーリエ変換したパワースペクトルなど

ビン数の多い
ヒストグラム表示が重い...
そういうものでしょうか？



今後考えていきたいこと

- 装置調整手順や試料測定手順（フロー）をうまく記述して自動化実験を楽に安全に行えるようなくみ（状態遷移の可視化、インターロックetc...）



[https://en.wikipedia.org/wiki/Scratch_\(programming_language\)](https://en.wikipedia.org/wiki/Scratch_(programming_language))

- これまでのご講演で特に刺さったトピック
 - SlowDash?
 - SPADI?