加速器制御について

その必要性と実現

古川 和朗

高エネルギー加速器研究機構 (KEK)

< kazuro.furukawa @ kek.jp >
<http://www-linac.kek.jp/linac/>
<http://www-linac.kek.jp/cont/>

Introduction

なんで制御なんて重要なの？

- 制御って計算機で装置情報を読み書きするだけでしょ
  - 部分的には正しい、しかし実際に加速器に運用すると課題に直面する

- 加速器制御の目的は加速器の実力を発揮させること
  - 加速器は複数の先端技術の複合装置として構成されており、それらを協奏的に動作させることができなければ期待する性能を発揮することができない
  - 研究用の大型加速器は基礎科学を目的とし、人間の飽くなき好奇心を満足するために利用され、対価が明確には存在しない、そのためにどの程度の人的資源を投入すべきかの指標が存在しない
  - 結果的に投入する人的資源を制限する方向に力が働き、効率的な運転制御システムが求められる。

なんで制御なんて重要なの？

❖制御に関わる自分達にとっては

◆加速器制御の開発や運用においては、全ての装置担当者や全ての運転担当者と情報交流が必要となる

◆結果として加速器全体に影響を与えることができる

まず始める前に

◆加速器の現場には

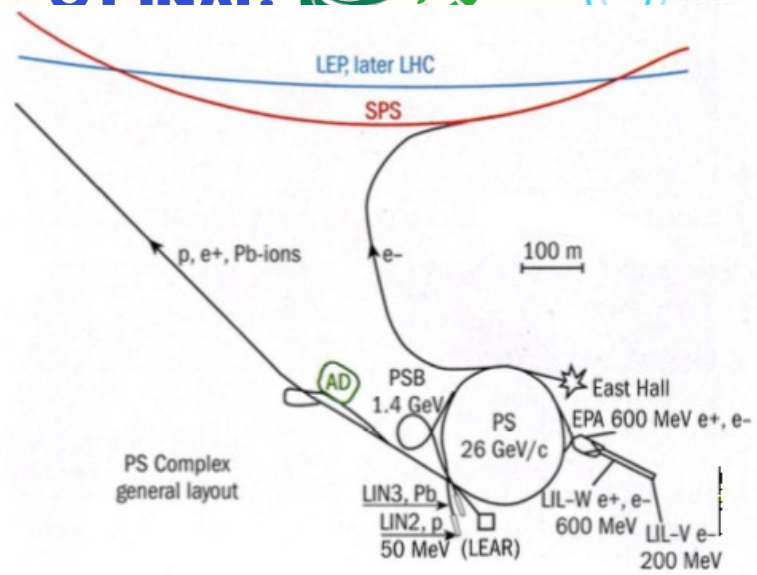
- さまざまな制御計算機や接続装置が配置されている、それらを有機的に協調動作させることにより加速器が性能を発揮する

◆40年以上前の加速器の制御は

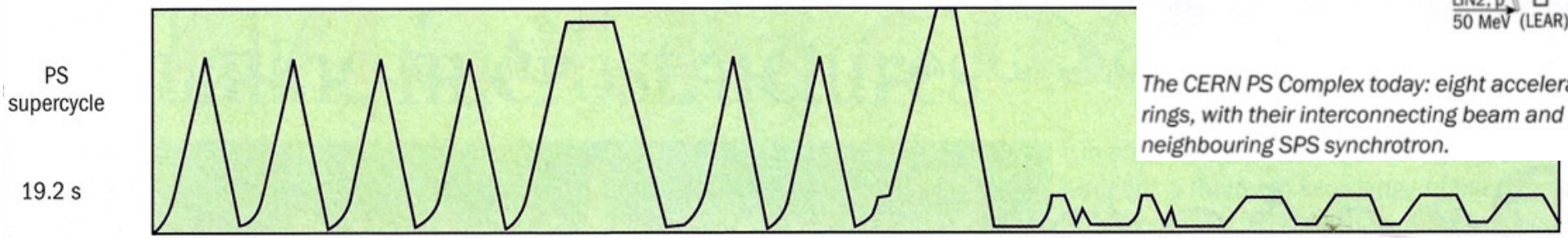
- 現場の装置と制御室のつまみや表示器が直接接続されていた
- 装置が多い場合は多数の切替器を駆使していた
- 徐々に電子的な接続装置に置き換えられた
- 計算機が設定値の記録に用いられるようになった
- **Unix, MS-DOS, Ethernet, TCP/IP**は使い物にならなかった
- 各研究所は異なる計算機や**OS**を使用し機能の共有は困難だった
- 一方、ビームシミュレーションや**A.I**もオフラインで利用され、将来のオンライン利用が期待されていた

PPM の夢

- **CERN PS Pulse-to-pulse modulation (PPM)**
  - **1.2 (2.4) 秒毎に異なる粒子の加速 (e-, e+, p, ṕ, HI)**
  - **KEK でも可能なのではないか**
    でもどうやって？



The CERN PS Complex today: eight accelerators and storage rings, with their interconnecting beam and transfer lines to the neighbouring SPS synchrotron.

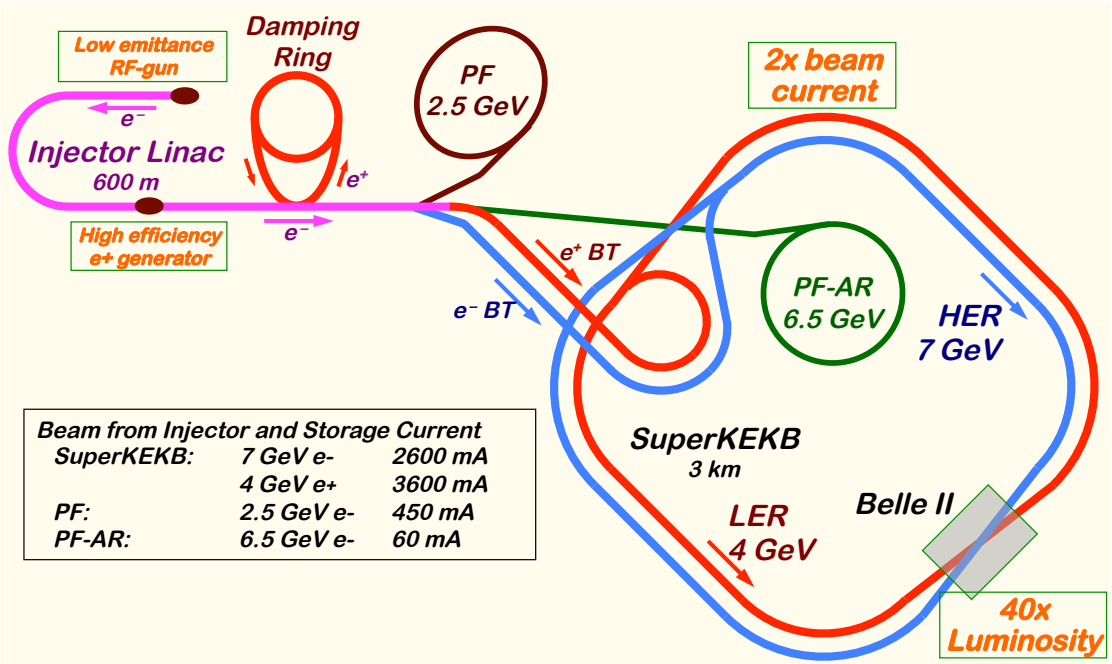


particle	S16+	S16+	S16+	S16+	p	O8+	O8+	p	$\bar{p}$	p	e+	e-	e+	e-
PS user	SPS	SPS	SPS	SPS	East Hall	PS SPS ion test	PS SPS ion test	AAC $\bar{p}$ production	LEAR	$\bar{p}$ transfer test	SPS LEP	SPS LEP	SPS LEP	SPS LEP

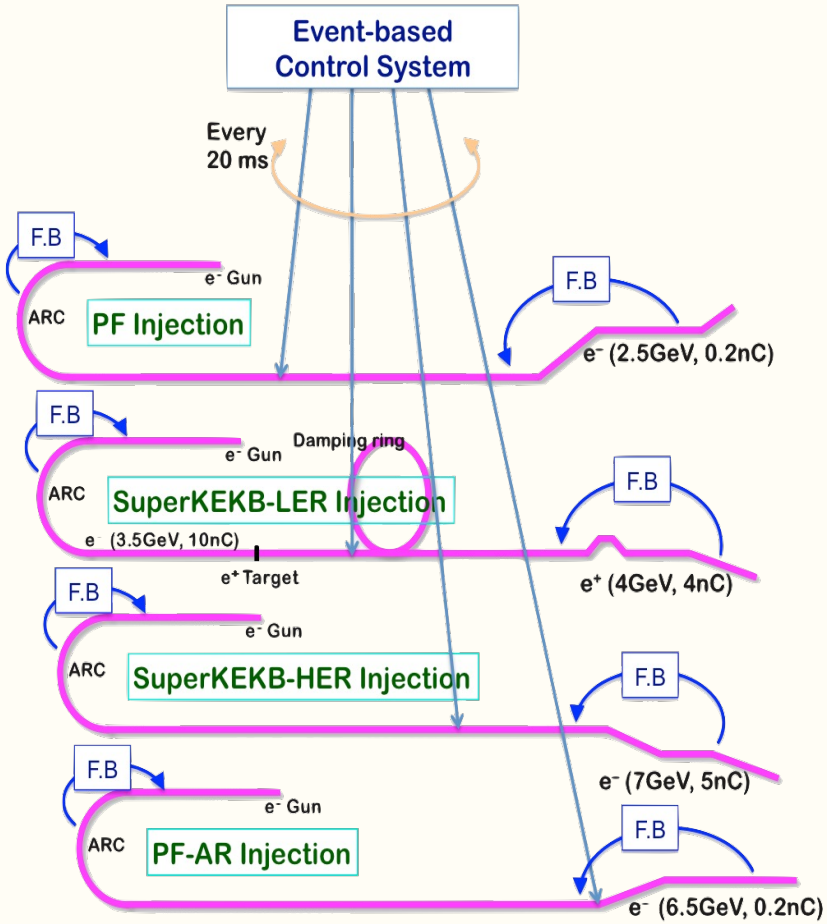
Fig. 1: A great variety of PS cycles are composed into a "supercycle", in which one user after another is served. In a particularly rich case, from August 1990, no less than six different kinds of particle (sulphur ions, protons, oxygen ions, antiprotons, electrons and positrons) were sequentially served to seven different destinations.

Mission of Electron/positron Injector in SuperKEKB

- For 40-times higher luminosity in SuperKEKB collider
- Low emittance & low energy spread injection beams with 4 times higher beam current
  - New high-current photo-cathode RF gun
  - New positron capture section
  - Positron damping ring injection/extraction
  - Optimized beam optics and correction
  - Precise beam orbit control with long-baseline alignment
  - Simultaneous top-up injection to DR/HER/LER/PF/PFAR
- Balanced injection for the both photon science and elementary particle physics experiments

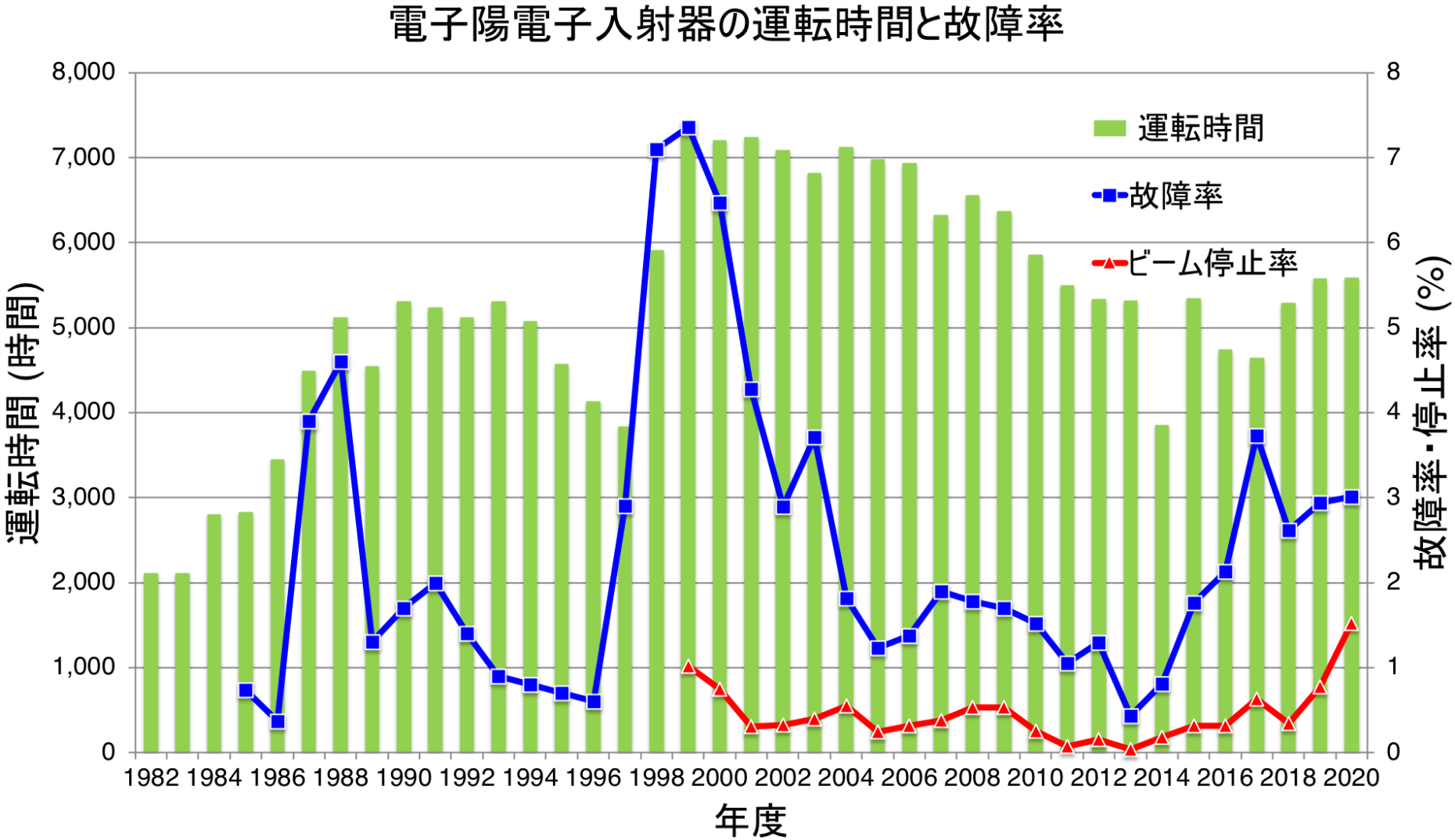


Beam from Injector and Storage Current		
SuperKEKB:	7 GeV e-	2600 mA
	4 GeV e+	3600 mA
PF:	2.5 GeV e-	450 mA
PF-AR:	6.5 GeV e-	60 mA



The single injector would behave as multiple injectors to multiple storage rings by the concept of virtual accelerator

電子陽電子入射器の故障率



- “故障率” は一部入射可能な場合も含む装置の故障を意味し、“ビーム停止率” は入射が不可能となる故障を意味する
- **TRISTAN, KEKB, SuperKEKB** プロジェクト開始時には新しい機器の故障率が上昇する（制御の問題も含まれる）
- さらに最近の同時入射と **SuperKEKB** のビーム品質要求の高さが運転を難しくしている

◆制御によって運転を改善できる余地があるかも

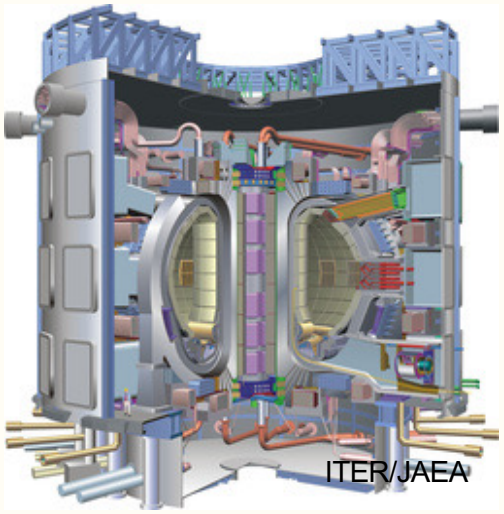
Overview

巨大科学と制御

- 巨大科学 (死語?) とかビッグサイエンスと呼ばれる分野の実験装置を用いて数十から数千の研究者が実験を行う
  - 高エネルギー加速器、光学・電波・重力波望遠鏡、トカマク・慣性核融合、などの大型物理実験装置
  - 国際協力により設計・建設・運用される傾向にある
  - ジュネーブのLHC/CERN, カダラッシュのITER, アタカマのALMA, つくばのSuperKEKB, ILCやFCCの国際共同開発


LHC/CERN


ALMA/NAOJ


ITER/JAEA

先端技術の民生化と民生技術の高度利用

- 研究用や特殊用途だったUnix、TCP/IP、組み込み計算機などが民生用にも利用されるようになり、共通化が図られると同時に信頼性が向上した
- 特にUnixとX Window (X11)を多数の研究所で利用することが現実的になり、研究所間でソフトウェアの交換・共通化が可能となった
- 加速器制御用ソフトウェアの共通化を目指した活動と会合が数多く開かれた
  - グループによって、共通ソフトウェアの定義から始めようとしたり、新規加速器のソフトウェアの設計を他の加速器にも応用可能なように考慮したりした
- それまで何度も行われてきた加速器制御の構造とソフトウェアの開発を統一して研究所間で共有し、もっと本質的な仕事に集中したいという思いがあった

加速器制御に関連する国際会議

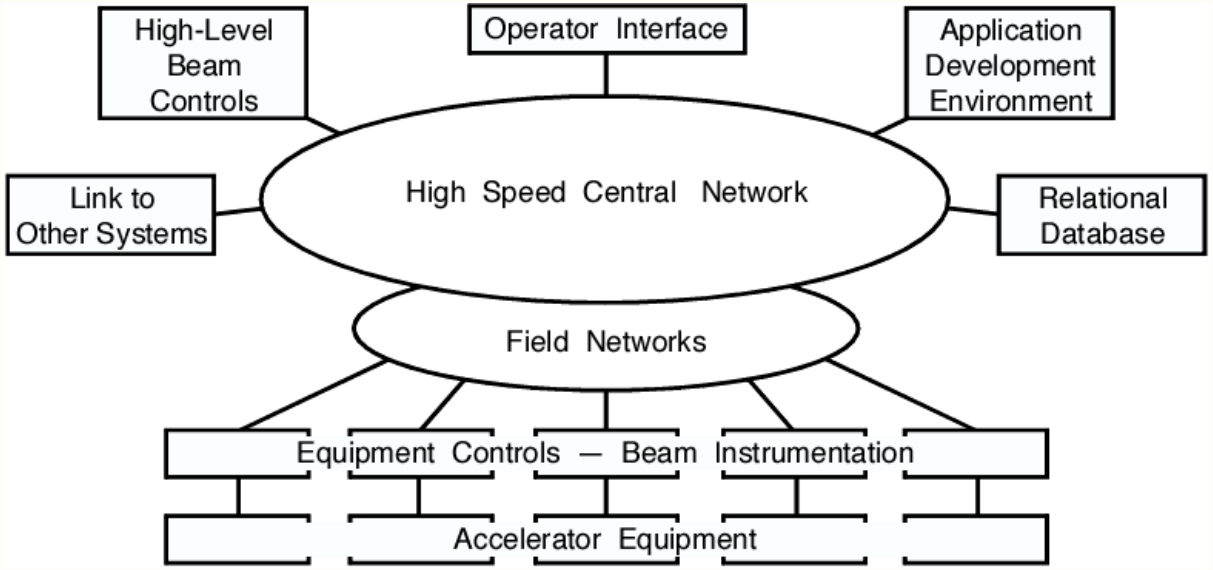
◆そこで、加速器設計や装置だけでなく、運転制御についても国際的に知見を共有し、同じ開発の繰り返しを避けようという気運が高まった (40 年前ごろ)

- ICALEPCS, International conference on accelerators and large experimental physics control systems
- PCaPAC, International workshop on emerging technologies and scientific facilities controls (International workshop on personal computers and particle accelerator controls)
- WAO, International workshop on accelerator operation

制御システムの目標 (の例)

◆信頼性が高くかつ柔軟な制御処理機構を道具として提供する

- ❖性能を高めるための (実時間) 自動処理
- ❖モデリング・シミュレーション
- ❖新しい問題に対する迅速な対処
- ❖情報の交換・管理
- ❖データ解析
- ❖安全機構



加速器制御の難しさ

◆目的仕様

- ❖制御仕様は加速器設計や装置設計が固まらないと決定できない
- ❖もちろん、最終目標は行われる実験の成果だが、制御能力の貢献が間接的
- ❖しばしば運転開始後の仕様変更が求められる
  - ⌑確実性とともに柔軟性が必要
  - ⌑迅速な試行錯誤能力も必要 **(Rapid prototyping)**
    - ◆日々新しいビームの運転方法が考え出されて運転方法が試されるけれども、ほとんどは捨てられる
- ❖加速器間で共有される技術も多いが、個々の加速器に固有の機能もまた多く、未だ制御機能の一般化は難しい
  - ⌑国際協力でアイデアの交換が継続されている

解決すべき課題

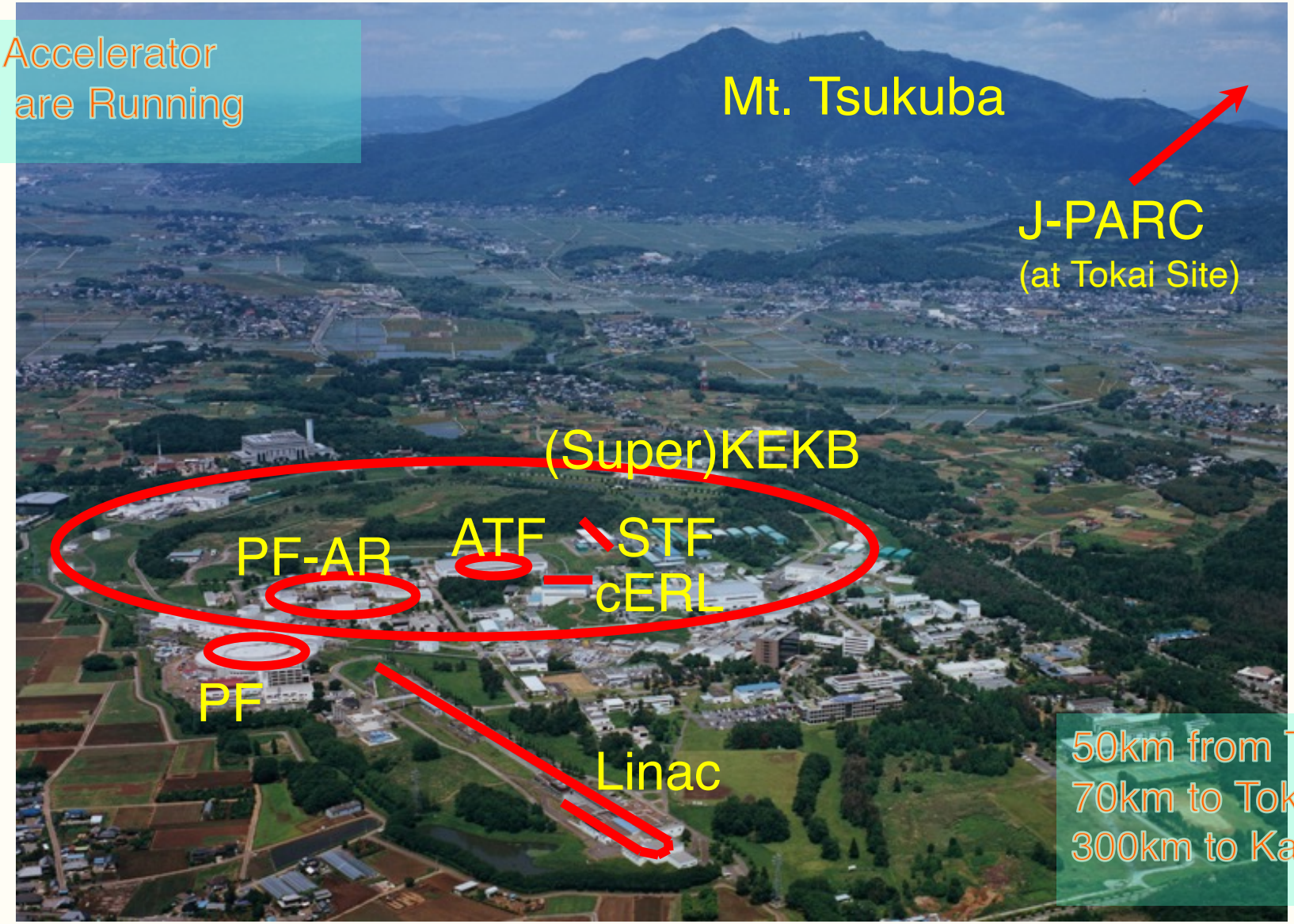
- ◆装置の数の多さ
- ◆装置の種類の多さ
- ◆要求仕様の収集の難しさ
  - ❖共通の理解のために加速器と装置についての知識が必要
  - ❖加速器設計グループや装置グループとの共通理解作り
- ◆産業用・民生用機器の有効利用
- ◆グループ間の相互協調が重要
  - ❖どのプロジェクトでもこれが足りなかったかも、という反省がある

解決すべき課題

- ◆オンラインビームシミュレーション機能の実装
  - ❖限られた時間で迅速な運転手続きの実装とそのフィードバックが必要
  - ❖予定されていない要求仕様の変化への追従も必要
- ◆並行して日々の運転への対応も
  - ❖日常的な手続きの自動化
- ◆パフォーマンスの分析能力
  - ❖可能な限り多くの履歴蓄積と履歴情報の分析機能
    - ⌑ 傾向、相関、突然の変化などの認識の手助け
  - ❖制御室だけでなく研究室からも情報解析
- ◆人や加速器装置、実験装置の安全保護機能
- ◆それぞれの機能は、個々の装置とビーム運転の双方に必要
- ◆個々の加速器に特化する部分と共通化できる部分の見極め

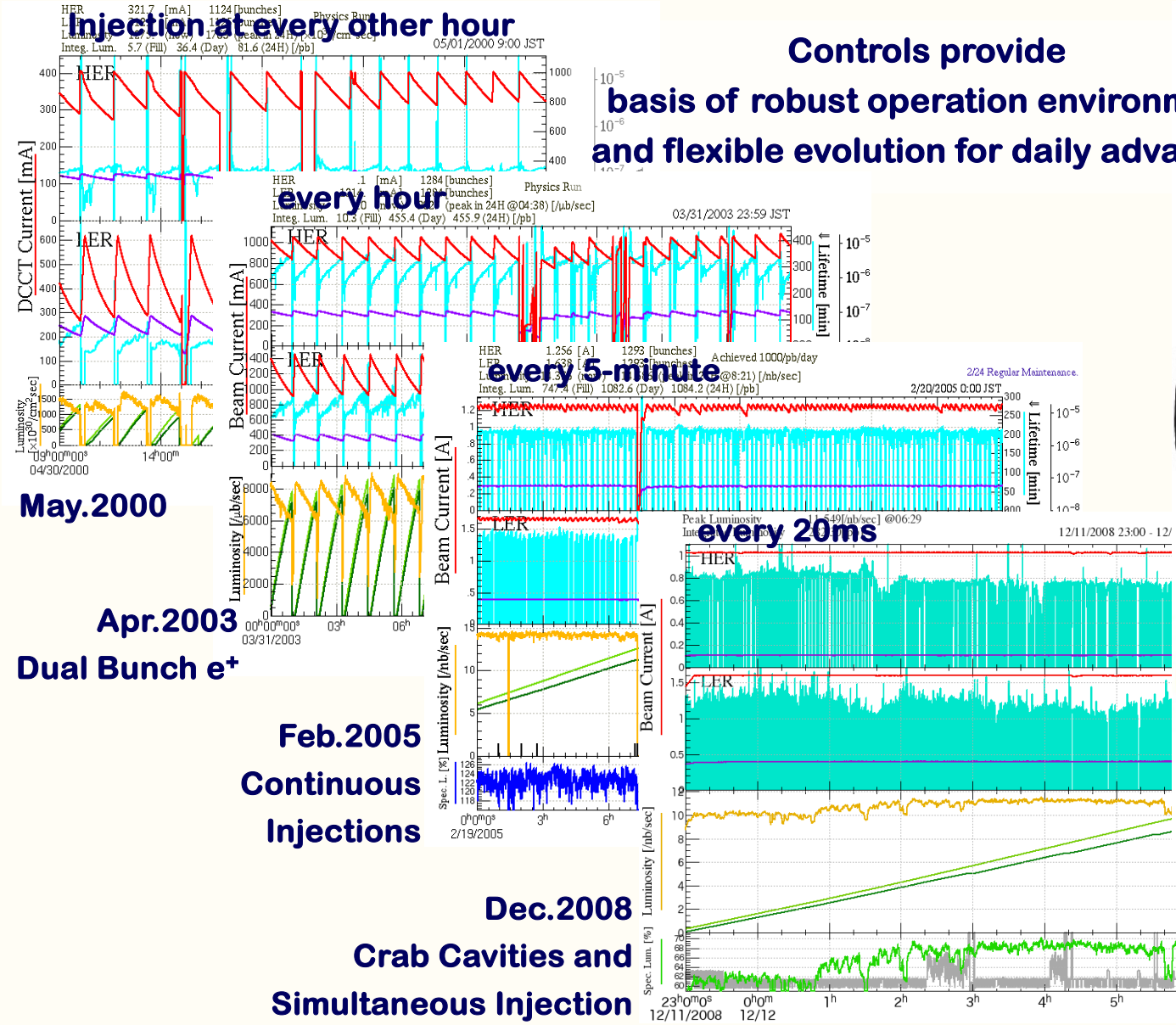
Accelerators at KEK

Several Accelerator Projects are Running

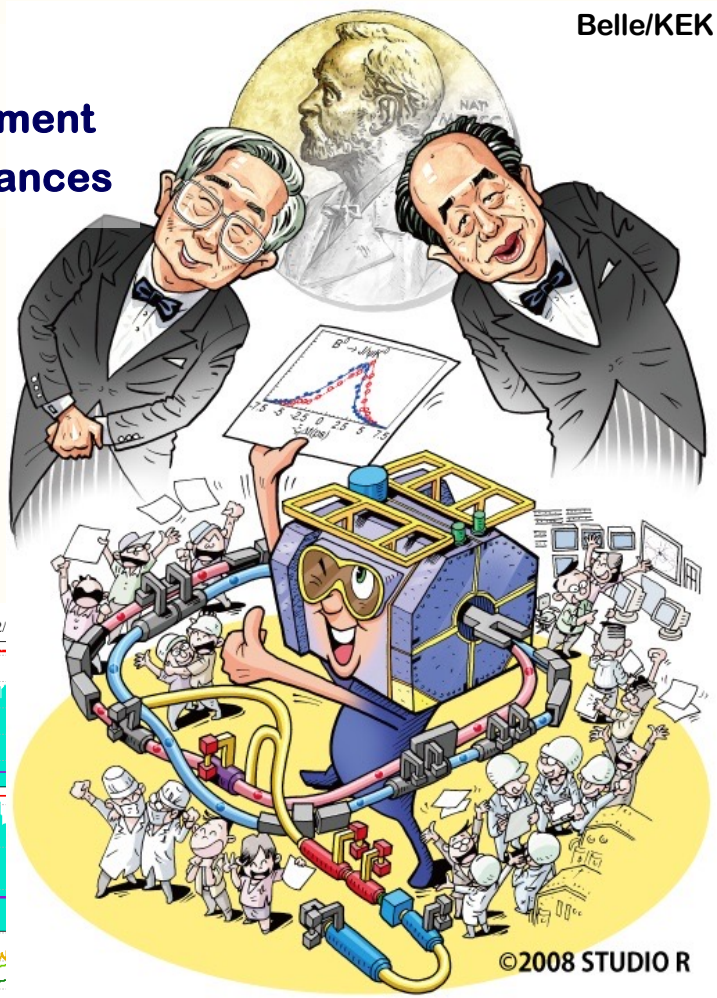


50km from Tokyo
70km to Tokai
300km to Kamioka

KEKB の場合の運転形態の変遷



Controls provide
basis of robust operation environment
and flexible evolution for daily advances



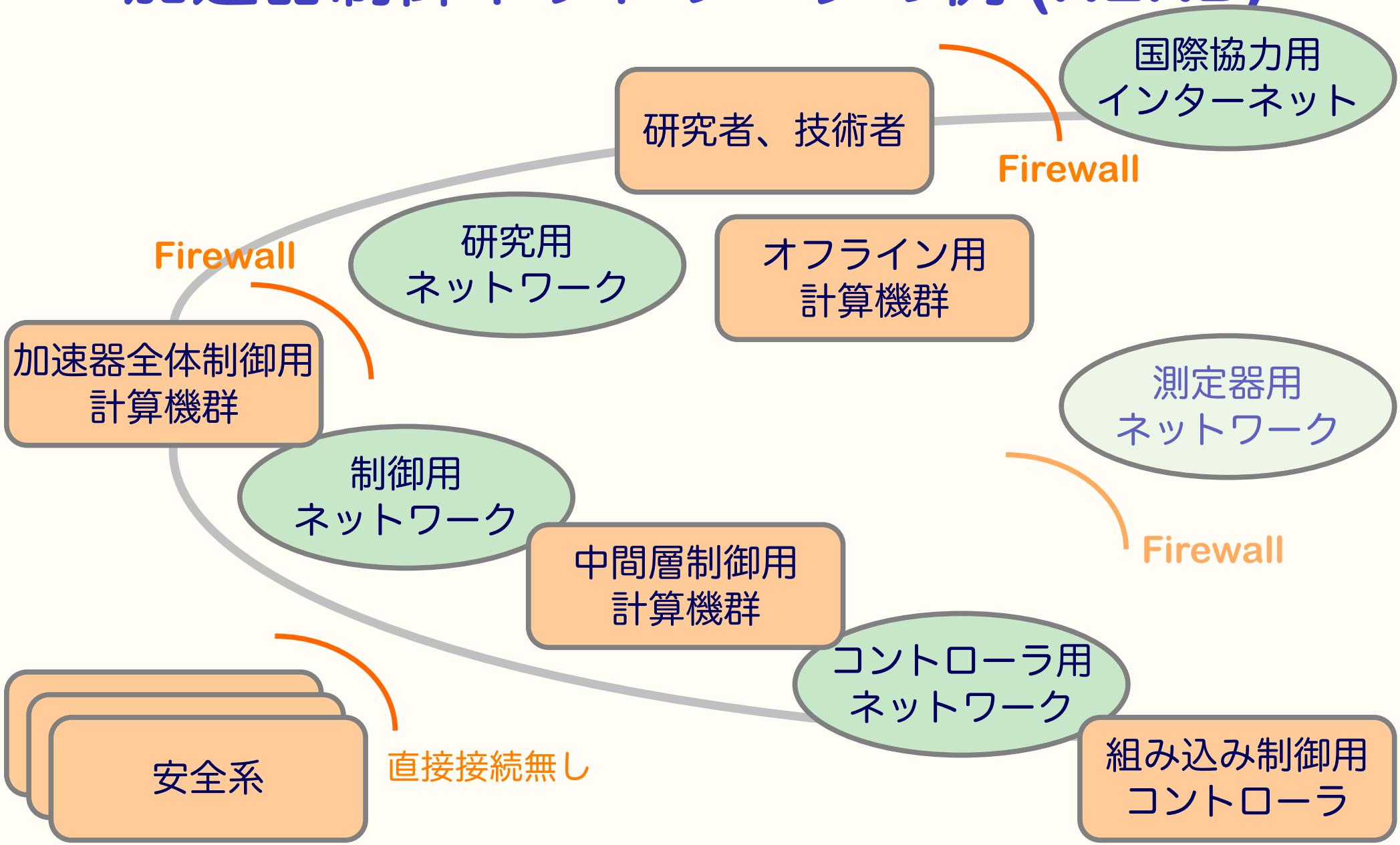
Important to follow
changing demands
flexibly

典型的な加速器 (SuperKEKB)

- 多数の装置の複合体
  - 数千の電磁石
  - 数十の大電力マイクロ波生成装置
  - 数千の真空装置
  - 数千の診断装置
  - 数百人の研究者
    - 現在では粒子加速器と測定器は分業している
    - 加速器も単なる電磁気学の応用だけでは無く物理学の研究対象
      - 化学の重要性は原子力に比べると低いので、見通しが立てやすい
    - また大規模超伝導利用や高真空、レーザーなどさまざまな分野と連携

- 制御
  - 数千の組み込み小型コントローラ
  - 数十～数百の中間層のサーバー
  - 数十の取りまとめ用のサーバー
  - 主に IP を利用したネットワーク
  - これらを協調させて加速器を運転指揮する (Orchestrate) 必要がある

加速器制御ネットワークの例 (KEKB)



Some of Available Technologies

PLC

- **Programmable Logic Controllers (PLC)**
  - Rule-based algorithms can be well-adopted for simple controls
  - IP network for the both controls and management were preferable
    - Especially at KEK/Linac which has a policy of IP only field network
  - ~150 PLCs at Linac since 1993, and also many at J-PARC
  - Isolated/separated development becomes easy
    - Outsourcing oriented
  - Equipment developer oriented
    - Many maintenance capabilities were implemented
  - IEC61131-3 Standards
    - 5 languages, with emphasis on naming
    - Effort to make common development environment
    - Should be paid more attention
  - Redundancy sometimes possible
  - Embedded Linux/EPICS possible
    - Efficient reuse of resources, cheap, quick, etc.

Network with only IP/Ethernet

- **The policy chosen when we upgrade Linac in 1993**
  - **Make network management simpler**
    - **Faster switches, routing, network-booting, etc.**
  - **Avoid Hardware failure and analysis effort with old field network**
    - **Home-grown field networks need much dedicated man-power**
  - **Cost for optical Ethernet went down at around 1995**
    - **Linac has high-power modulator stations, noise source**
  - **Nowadays many facilities have this policy with GbE**
    - **J-PARC controls basically followed this**
  - **More and more intelligent network devices**
    - **ex. Oscilloscopes with Windows/3GHz-Pentium built-in**
    - **Even EPICS IOC, MATLAB, or others can be embedded**
  - **Network components can be replaced one-by-one**
  - **Security consideration will be more and more important**

FPGA

- Another "everywhere" after IP network
  - Digital circuit and software can be embedded into one chip
    - Even CPU core, Linux, and EPICS are embedded
    - Flexible and robust, wonderful platform for local controls
      - Sometime terrible source of bugs
  - Nano-second level timing
  - More and more gates, memory, pins, etc
  - More software support needed
  - Open hardware initiative may enhance the benefits

ATCA and μTCA

- **Advanced telecommunications computing architecture**
  - Accommodate many 100ohm serial buses
  - GbE or PCI-express, 10GbE, etc
  - Typically 14slots in 19” width and 12-unit height
  - Shelf manager manages healthiness of the system
    - through Intelligent Platform Management Interface (IPMI)
  - Many reliability improving facilities, redundancy, hot-swap, etc.
- **MicroTCA**
  - More recently defined in 2006, based on AdvancedMC Mezzanine Card defined in ATCA
  - Begin to acquire many facilities from ATCA
- **Future accelerators**
  - Development started for ILC, XFEL/DESY, and detectors

EPICS

EPICS

- たくさんの制御システムが同じ問題を抱えていたのに、違うハードウェア/ソフトウェアを使い違った方法で解決しようとしていた
- そこで EPICS が開発された
- ほとんどの制御についての要求が満たされる
  - Closed Loop / Open Loop
  - Logical Value Conversion
  - Synchronous / Asynchronous
  - Programless / Performance
  - Display / Archive / Analysis

制御ソフトウェアツールキット EPICS

- 以前はほとんどが内製ソフトウェアの組み合わせ
  - RPC, CORBA, …
- 徐々に共通ソフトウェアに移行してきた
  - KEK では 1994 年から “EPICS” と呼ばれる国際共同開発のソフトウェアへ
    - BSD-like な Open-source、多数の研究所が採用
    - SCADA や Labview などとほぼ同等またはそれ以上の機能
    - 他に CORBA を基本としたものなどもある
      - 未だもってネットワークワイドな OS が存在しないのは残念
  - 当初は中間層と全体制御の間で EPICS を利用していた
  - 現在は小型コントローラにも内蔵するようになってきている
    - ITER (プラズマ核融合) も採用している
- 上位ソフトウェアはスクリプト言語で書かれるものも多い
  - 粒子加速器の物理の記述が行えて Mathematica 文法に近い “SADscript” というソフトウェアが多用される
  - 他に Python や Perl、Tcl なども
- 下位ソフトウェアは C++ や C が多い

共有可能な制御技術

- 1980年代後半 CERN や複数の米国の研究所でそれぞれ共有可能な制御機構が提案された
- LANL が SDI/GTA 向けに開発した EPICS が新設の ANL/APS に採用されて整備された後、SSC に採用され、さらに SSC 自体が中止になったことから、米国内で広まることになった
  - 一時期 VxWorks を利用することが多かった天文台施設にも複数採用された
- 当初 OS としては SunOS と VxWorks のみだったものの、2000 年ごろには OS に縛られなくなった
- データベース駆動で汎用性高く設計されていながら、速度も失わないよう注意が払われていた
- どの制御システムでも必要な基本機能を再度開発することを避けたい、という思いはどの研究所も同じ

EPICS

◆EPICS を使用すると

- ❖これまで各研究所で作られていた制御のためのプログラムを個別に書いたり、保守したりする必要がなくなる
- ❖よりよく考慮された設計になっている可能性が高い
  - ⌑当初は考えていなかった制御が必要になることは多い
- ❖多様な制御方式に対応できる
- ❖多様なアプリケーションプログラムを使用できる
- ❖他のグループからの支援を得られる可能性がある
- ❖国際的に貢献できる可能性がある
- ❖x 最新のソフトウェア技術の利用が制限される
  - ⌑オブジェクト指向プログラミング、CORBA、とか
  - ⌑部分的に使用することは可能

EPICS

◆他の可能性

- ❖自分独自の制御ソフトウェアと制御方式
  - ⌑ 40 年前は普通だった
  - ⌑ それ自体を研究対象にしたいならいいが、現在では非現実的
- ❖TANGO
  - ⌑ CORBA を利用しており、これからの制御システムの方向性を示している可能性はあるが、各研究所で作り込まなくてはならない部分が多く、我々がすぐに利用できる部分は多くはないかも
- ❖CERN
  - ⌑ RPC, Java-CORBA から変遷している。測定器と施設と加速器が協力しているが、それぞれ独自の使い方をしている。LHC は規模が大きすぎ、他の研究所が利用できるか不明。FAIR/GSI が採用している
- ❖LabView
  - ⌑ 広く使われているが、大きなシステムを作る場合に、設計、保守等問題点の解決が必要
  - ⌑ 部分的に EPICS と組み合わせて使っている研究所は少なくない

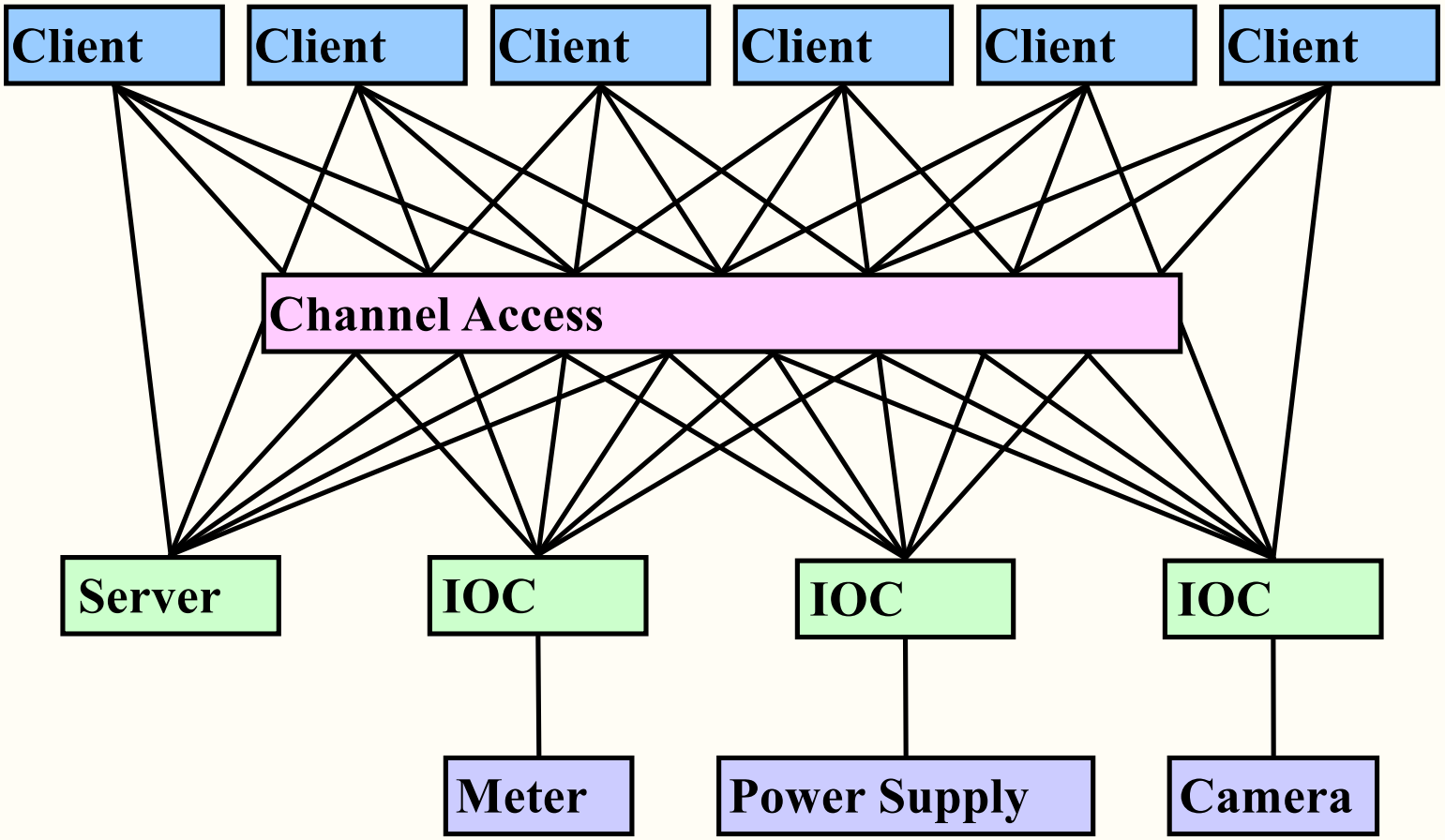
I/O Controller (IOC)

- Database driven な EPICS の Server
- 約 30 の基本 Record type とユーザ定義 Record
  - Ex. Analog Input (ai), Calculation (calc), etc
  - PID 制御 (epid) という Record もある、Link するだけで Closed loop feedback を構成できる (現実にはいろいろ考慮が必要)
- 個々の Record はそれぞれに数十の Field を持つ
  - Ex. 名前, 変換規則, Alarm, Deadbands, Simulation, Links, etc
- Record の Instance が Process variable (PV)
- 一つの IOC に数万の PV があっても良い
- 個々の PV は Database 指定により、定期的、割り込み時、Link 先動作、などで処理が起こり、必要があれば接続先に報告が送られる

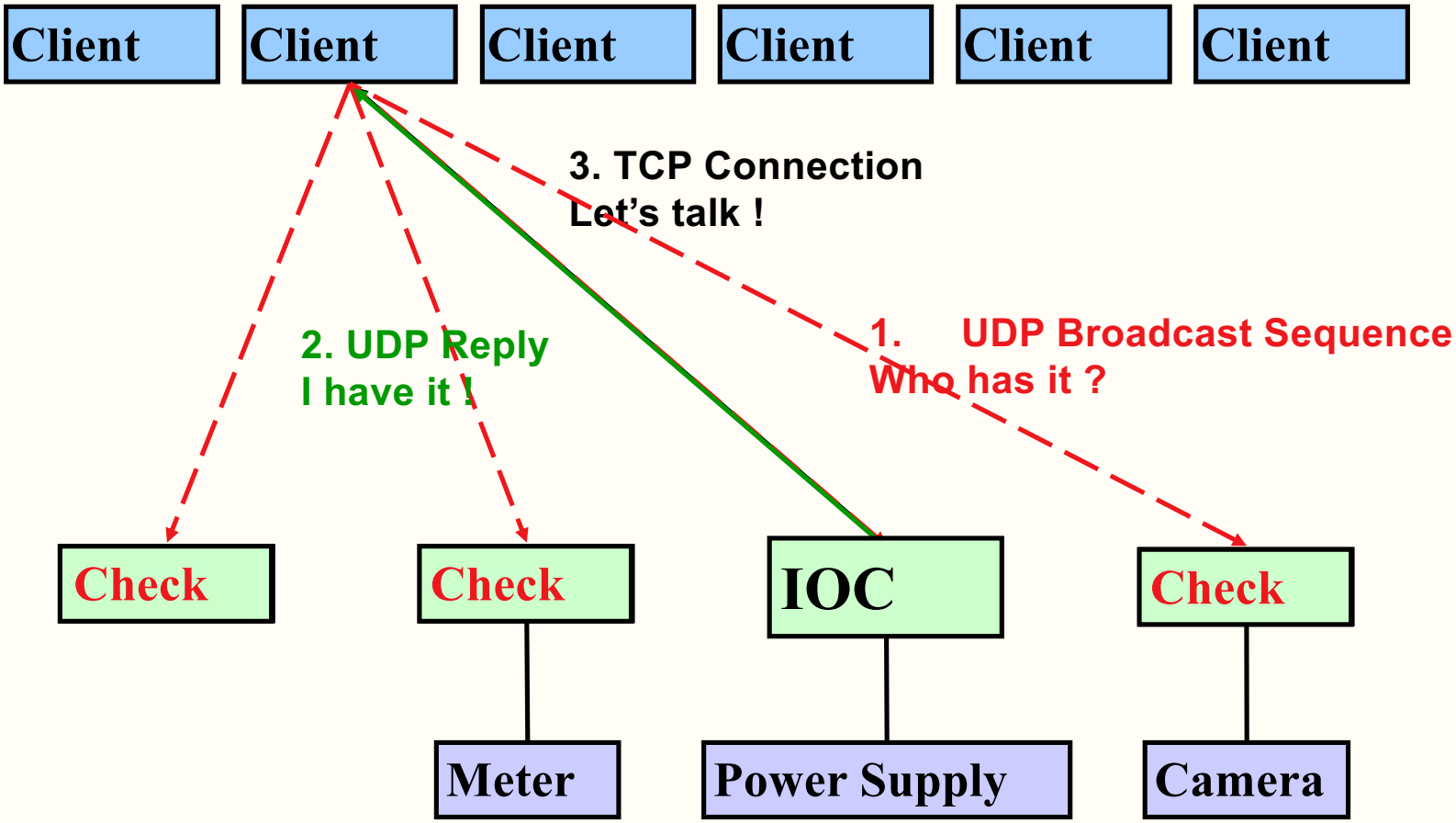
I/O Controller (IOC)

- Record/PV をユーザの Hardware に接続するのが Device support
  - 最も簡単には Hardware から値を取って Record support に返す数行の Code を書けば良い
  - 他の研究所で書かれた Device support を使っても良い
- 必要があれば自分独自の Record support を書く
- PV が動けば、後はたくさんの EPICS の仕組みを利用できる
  - PV link だけでもそれなりの Program が組める
  - さらに Sequencer などの簡易 Program 支援
  - そして、C/C++ による処理
- これらが IOC 内、または Channel access を通して IOC 間で利用できる
- さらに IOC と OPI (Client) 間にも世界が広がる

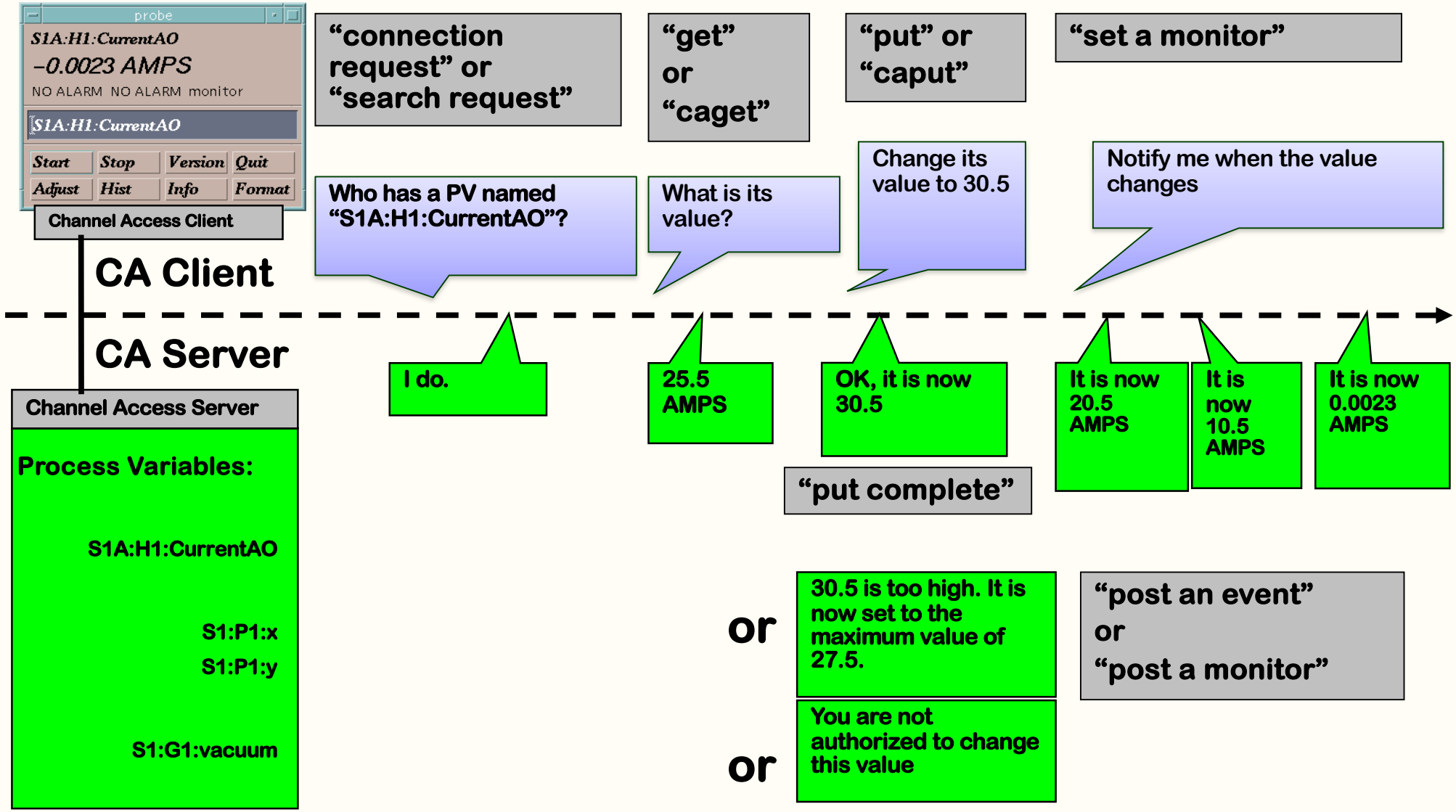
EPICS システムの構成



Search and Connect の手続き



Channel Access の動作



簡単な Code 例

```
/*caSimpleExample.c*/
#include <stddef.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "cadef.h"
main(int argc,char **argv)
{
   double    data;
   chid      mychid;
   if(argc != 2) {
        fprintf(stderr,"usage: caExample pvname¥n");
        exit(1);
   }
   SEVCHK(ca_task_initialize(),"ca_task_initialize");
   SEVCHK(ca_search(argv[1],&mychid),"ca_search failure");
   SEVCHK(ca_pend_io(5.0),"ca_pend_io failure");
   SEVCHK(ca_get(DBR_DOUBLE,mychid,(void *)&data),"ca_get failure");
   SEVCHK(ca_pend_io(5.0),"ca_pend_io failure");
   printf("%s %f¥n",argv[1],data);
   return(0);
}
```

注意：

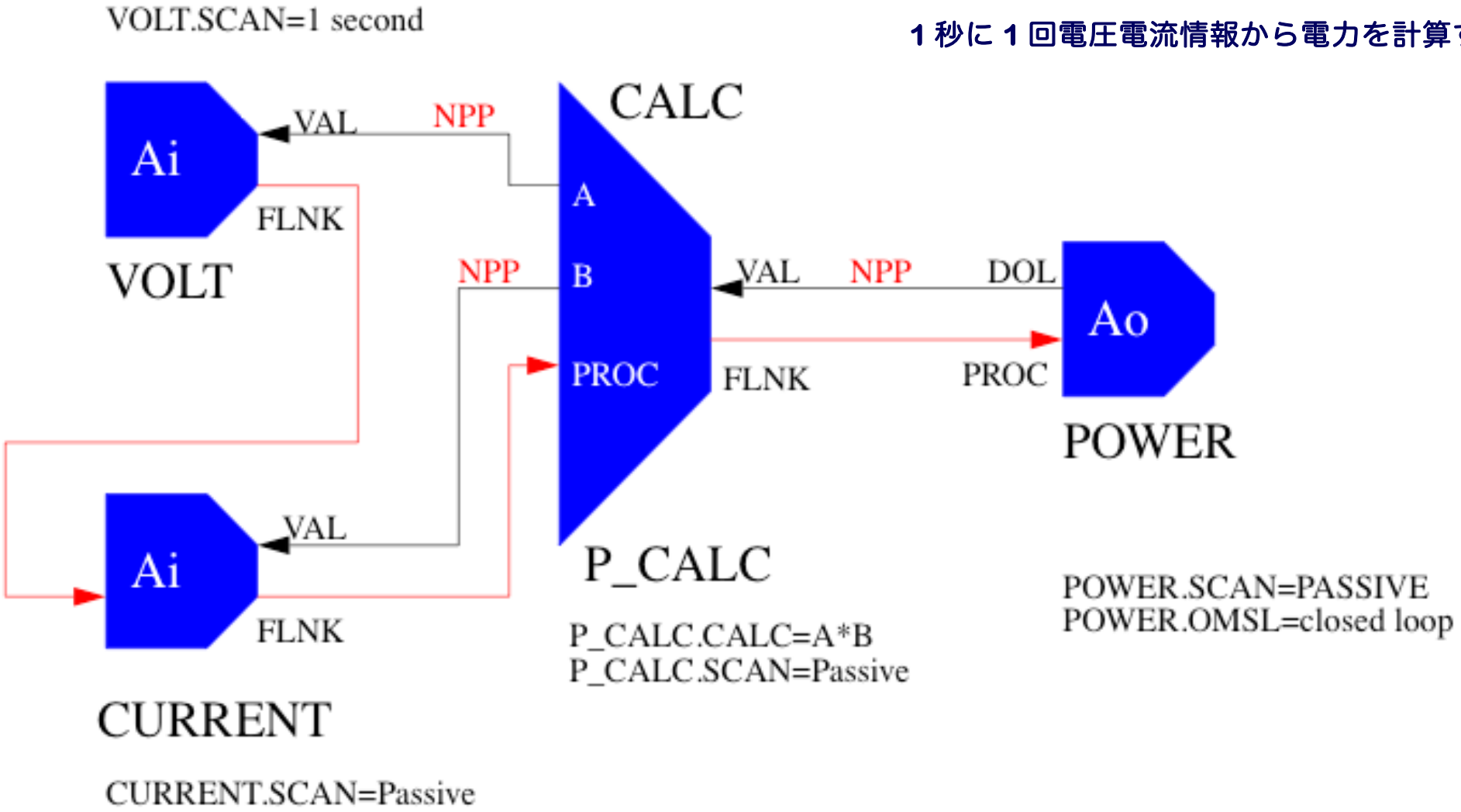
古い EPICS 3.13 の記法

エラー処理無し

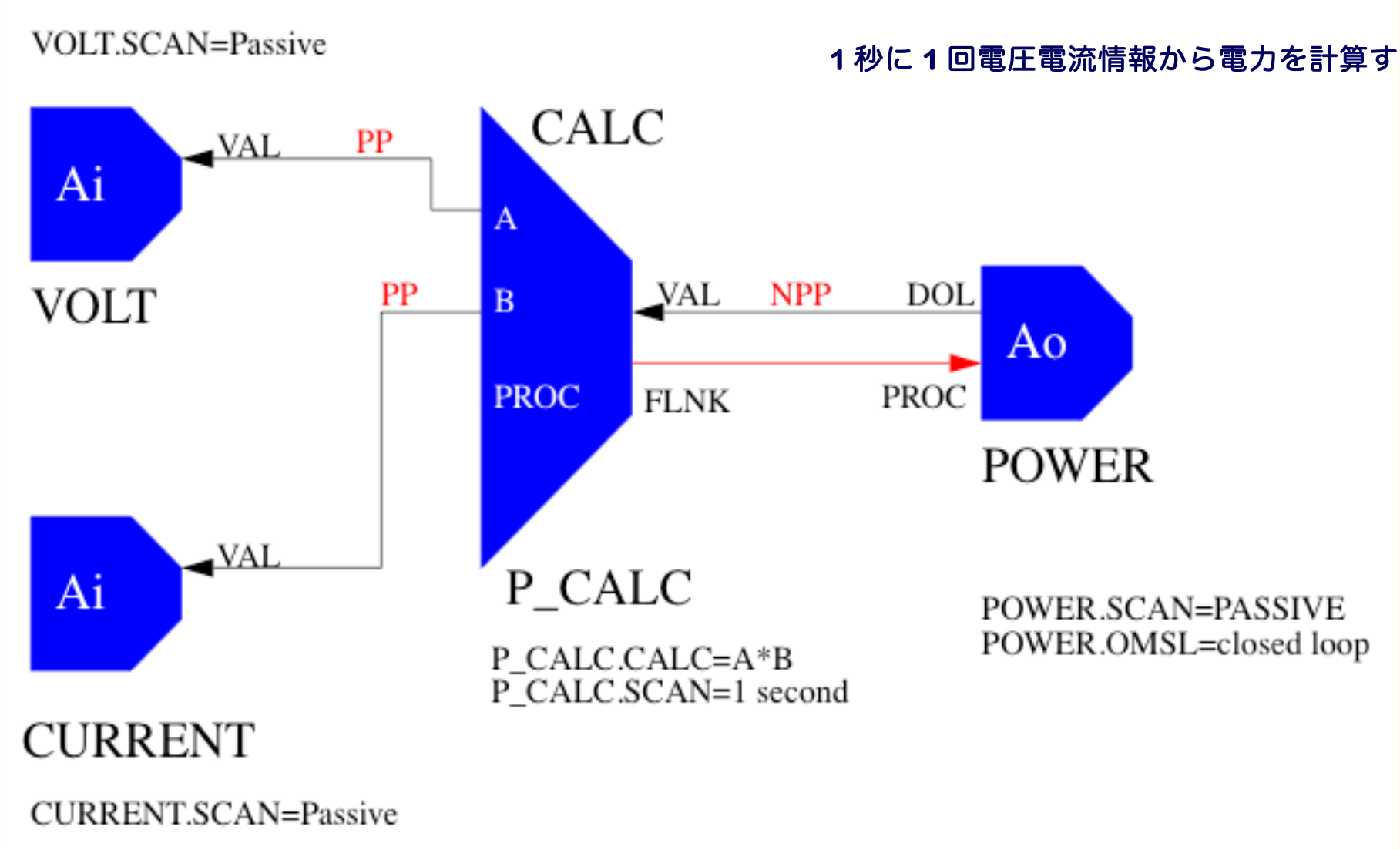
理想的な CA client?

- **Use callbacks for everything**
  - no idle 'wait', no fixed time outs.
- **Upon connection, check the channel's *native* type (int, double, string, …)**
  - to limit the type conversion burden on the IOC.
- **… request the matching DBR_CTRL_<type> *once***
  - to get the full channel detail (units, limits, …).
- **… and then subscribe to DBR_TIME_<type> to get updates of only time/status/value**
  - so now we always stay informed, yet limit the network traffic.
  - *Only subscribe once*, not with each connection, because CA client library will automatically re-activate subscriptions!
- **This is what EDM, archiver, … do.**
  - Quirk: They don't learn about online changes of channel limits, units, ….
    Doing that via a subscription means more network traffic, and CA doesn't send designated events for 'meta information changed'.

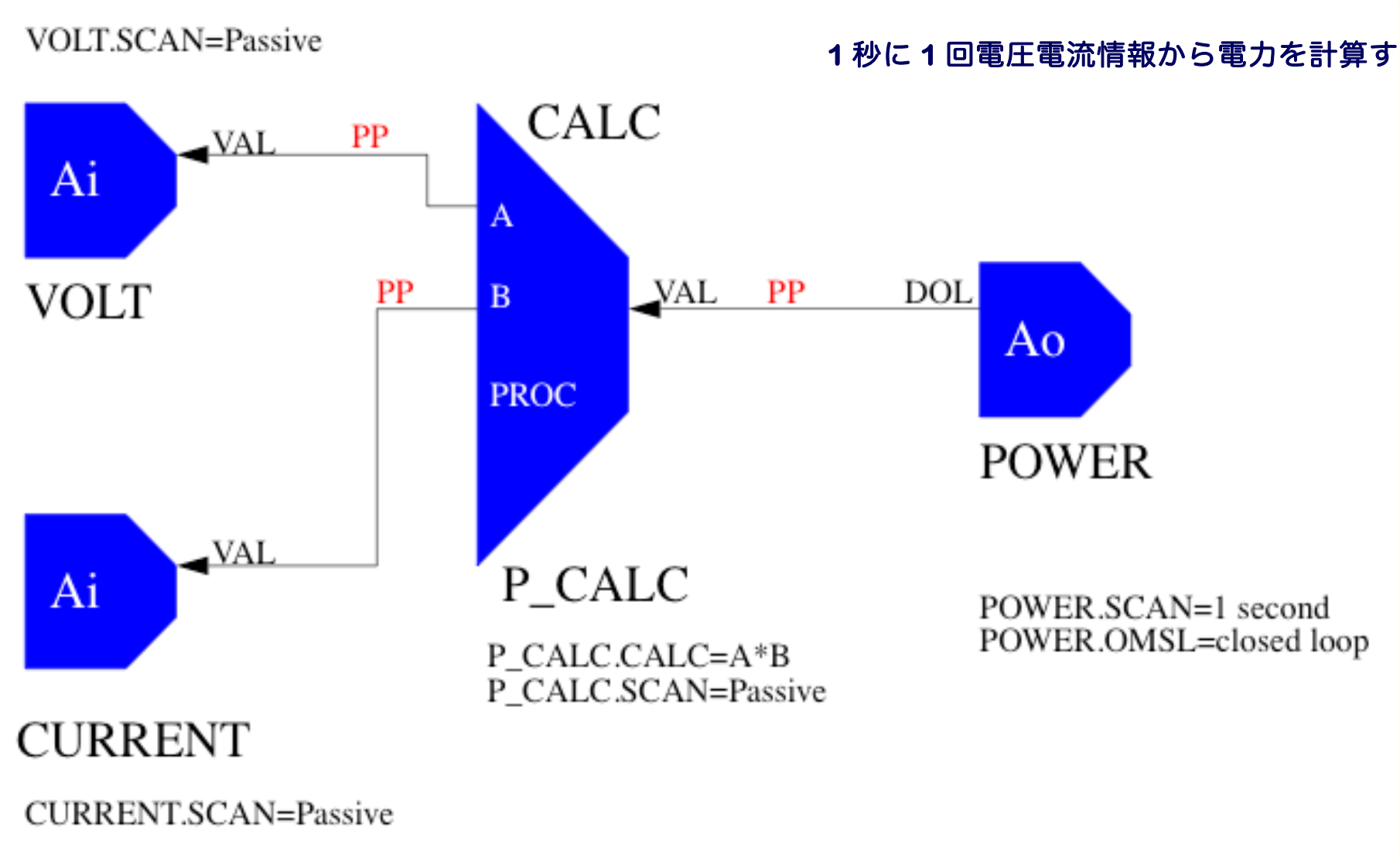
EPICS realtime database の動作



EPICS realtime database の動作



EPICS realtime database の動作



簡単な Device Support 例

```
struct {
        long            number;
        DEVSUPFUN       report;
        DEVSUPFUN       init;
        DEVSUPFUN       init_record;
        DEVSUPFUN       get_ioint_info;
        DEVSUPFUN       read_ai;
        DEVSUPFUN       special_linconv;
} devAiSecond = {
        6,
        NULL,
        init_ai,
        init_record,
        NULL,
        read_ai,
        NULL
};
epicsExportAddress(dset,devAiSecond);
```

注意：
同期処理のみ
エラー処理無し

```
static long init_ai(int after)  {
  return(0);
}

static long init_record(struct aiRecord *pai)  {
  pai->udf = FALSE;
  return(0);
}

static long read_ai(struct aiRecord *pai)  {
  time_t t;
  struct tm *tm;

  time(&t);                    /* get unix time */
  tm = localtime(&t);          /* convert into normal
clock time */
  pai->udf = FALSE;
  pai->rval = tm->tm_sec;      /* extract “second” data
*/
  return(0);                   /* convert */
}
```

EPICS

- Now is a kind standard, but …
- Object-oriented design support is limited now
  - Naming scheme, and/or design of new record
  - More software-engineering support favored
    - Several different efforts to provide better environment
      - Java IOC (M. Kraimer), Control system studio (M. Clausen), Data access (R. Lange)
- Security mechanisms
  - User, Host-based protection available
  - More security
    - Dynamic controls of security
    - Access logging/auditing, authentication, transaction mechanism
- Dynamic configuration of database
  - Dynamic creation / loading of records
  - Dynamic removal of records
    - Maybe some part of the codes can be shared with redundant-IOC project

Reliability and Availability

信頼性

- The end user expect rigid reliable operations
- Inner layers need flexibilities
  - Because of daily improvement
  - Compromise between
    - Practical or ideal solutions
    - Aggressive and conservative
    - Under restrictions of
      - Time, safety, budget, man-power
  - Here we think about adaptive reliability for 99.999% availability

hardware
hardware Interface
equipment controls
beam controls
linac
ring
accelerator physics
beam delivery
detector
data acquisition
computing
physics, chemistry, medical treatment

大きな犠牲なく信頼性を向上させるには

- **There should be “right way”**
  - We hope to have it some day, but for now we need interims
- **Surveillance for everything**
  - Well-arranged system does not need this, but…
- **Testing framework**
  - Hardware/Middleware tests just before Beam
  - Software tests when installed
- **Redundancy**
  - In Many Hardware/Software components
  - Of course some of them are Expensive, but…

さらなる信頼性の構築

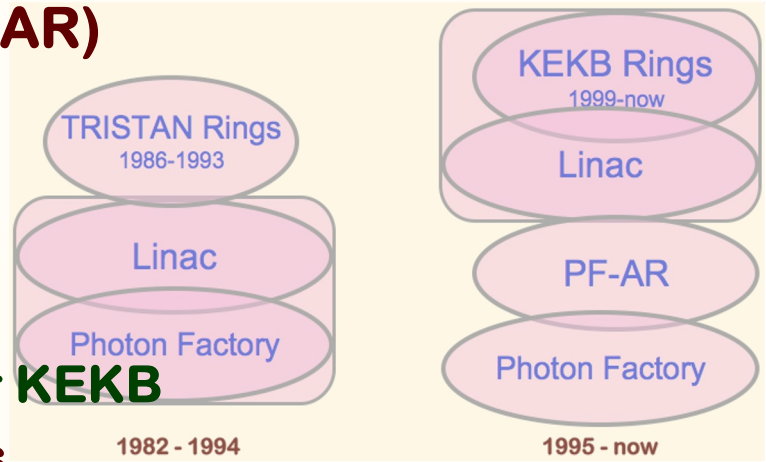
- **Well-defined software domain for accelerator controls**
  - It surely reduces the coding errors
    - But one may insist on a complete class of accelerator
      - which we can never achieve
- **Well-arranged naming conventions**
  - It surely reduces human-operation errors
  - And enables more computer-aided tools
    - But real world was not so simple
- **Well-specified deployment procedures**
  - Enables more computer-aided tools
  - Even scripting languages were said to be bad
    - But we are lazy

KEKにおける制御の実際

KEKB and Linac Control Systems

Linac

- Controls Upgrade (1990~)1993
  - De-facto (and International) Standards, IP-only Networks
- No long Shutdown for KEKB/SuperKEKB upgrade
- Gradual Transition towards EPICS
- Three indirect User Facilities (KEKB, PF, PF-AR)
- Fewer resources



KEKB-SuperKEKB

- 5 and 7-year Shutdown between projects
  - Precision requirements were much different for KEKB
- More or Less Complete transition of Controls
  - from Nodal at TRISTAN to EPICS+SAD at KEKB
- Basically Single-user (Belle)

制御システムの実装

- ◆5-10年毎に構成を再構築してきた
- ◆利用するハードウェア、ソフトウェアも過去を少しずつ引きずりながら、全てを失わないよう注意して、変化している
- ◆次の世代の技術を見つけることは容易ではないが常に探し続ける必要があるだろう
- ◆適正な次世代を見つけるには、現在の技術とその背景の理解が必要になる

Equipment Controllers at Linac

- 1982~(1997) (1st generation)
  - 300 microprocessor-based controllers
    - Linked together with home-grown fiber-optic network
- 1993~now (upgrade of controls)
  - 150 PLCs (programmable logic controller)
    - Linked via only Fiber-optic Ethernet/IP
      - Control communication with servers and program development
- 1995~now (upgrade for KEKB)
  - 30 VXI for rf measurement
  - 5 VME / 10 CAMAC for Timing
  - 20 VME for Beam monitors
- 2006~ (upgrade of BPM readout)
  - 24 Oscilloscopes with WindowsXP IOC for 100 BPMs
    - 10Gs/s, 50Hz acquisition, local processing with 20 calibration parameter/BPM

Equipment Controllers at KEKB

◆TRISTAN

- ❖Mostly CAMAC
  - ⌑ Equipment group responsibility: CAMAC module and outside

◆KEKB

- ❖100 VME/IOC without Analog processing
- ❖200 VXI/MXI mainframes for 900 BPMs
- ❖50 CAMAC crates are kept for rf and vacuum
- ❖ARCNet boards for Magnet ps. settings, and others
- ❖GPIB for Magnet ps. readback, and others
- ❖PLCs for Magnet interlocks, and others

EPICS Transition at Linac

- Home-grown RPC at Linac (1990~)
  - Bad timing but no choice because of end of old mini-computer support
- LynxOS Transition was developed (1994~1996)
  - To cover both RPC and EPICS with pthread, posix
    - Mostly working, Failed to get funding for Hardware/Software upgrade
- Gateways to EPICS in several ways
  - Software-only IOC and Gateway (Clients to both RPC/CA)
  - Portable Channel Access Server of EPICS-3.12 (1995~)
  - Soft-IOC with device support to Linac RPC (2002~)
- Real IOCs are increasing
  - PLC(rf,vacuum,magnet) and Linux, Oscilloscope(bpm) with Windows, VME(llrf and timing)
  - RPC servers read EPICS IOCs, EPICS gateways read RPC servers

EPICS Transition at KEKB

- Some candidates discussed after Nodal at TRISTAN
  - RPC/CORBA based control design
  - Reflective memory (hardware shared memory) design
- No other choice than EPICS for KEKB
  - No man-power for control system software
  - The choice at SSC
  - International collaboration was attractive

SuperKEKB に向けての検討

(現実は単純ではないけど)

KEKB の制御 1998 - 2010

- **EPICS を主要なソフトウェア機構として採用**
  - 既に 1995 年ごろには **de-facto standard**
  - いくつかの **fieldbus** が利用された
    - **VME, VXI, CAMAC, ArcNet, GPIB, etc**
  - ソフトウェアの設計試験の工数を大幅に削減した

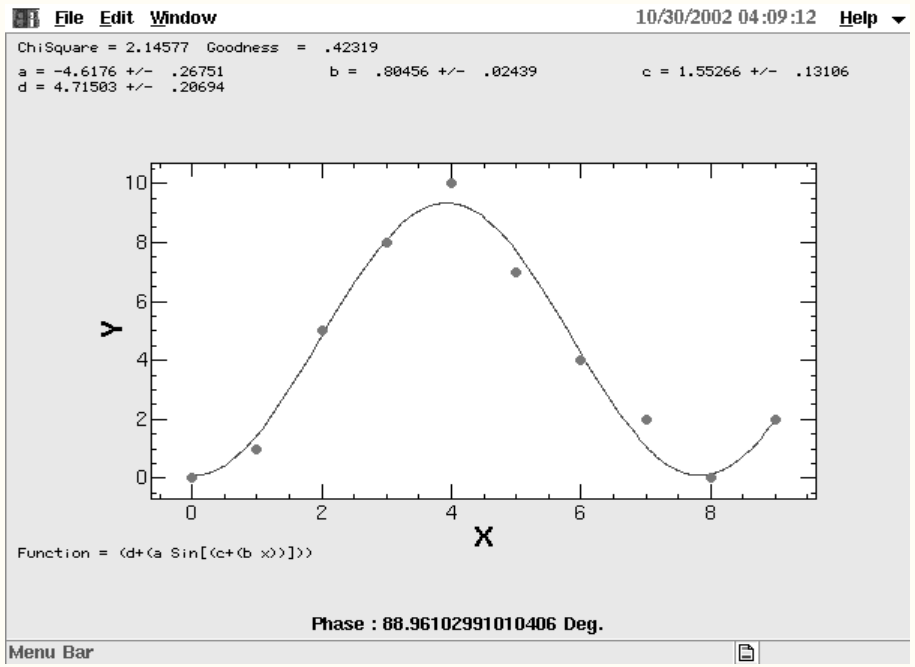
- **Scripting 言語が運転用に多用**
  - **SADscript/Tk, Python/Tk, Tcl/Tk**
    - 特に **SADscript** が、加速器ビーム制御、数値計算、視覚化 **(Data plot)**、画面制御 **(GUI)**、**EPICS** を接続した
  - この組み合わせを使用して、朝の打ち合わせで提案された新しいビーム調整方法を夕方には実現して運転を一変させるということが可能になった

SADScript

- **Mathematica-like Language**
  - Not Real Symbolic Manipulation, as a result rather Fast
  - EPICS CA (Synchronous and Asynchronous)
    - CaRead/CaWrite[ ], CaMonitor[ ], etc.
  - (Oracle Database)
  - Tk Widget
  - Canvas Draw and Plot
  - KBFrame GUI on top of Tk
  - Greek Letter
  - Data Processing (Fit, FFT, …)
  - Inter-Process Communication (Exec, Pipe, etc)
    - System[ ], OpenRead/Write[ ], BidirectionalPipe[ ], etc.
  - Full Accelerator Modeling Capability
  - Also Used for non-Accelerator Applications
  - Comparable to MAD, Elegant, XAL, etc. but very different architecture

SADScript

◆ Example




```
FFS;
w=KBMainFrame["w1",fm,Title->"t1"];
$DisplayFunction=CanvasDrawer;
W1=Frame[fm];
c1=Canvas[w1,Width->600,Height->400,
 Side->"top"];
Canvas$Widget=c1;
data = {{0,0}, {1,1}, {2,5}, {3,8}, {4,10}, {5,7}, {6,4}, {7,2}, {8,0}, {9,2}}
fit = FitPlot[data,a Sin[x b + c] + d, x, {a,5},{b,1},{c,1},{d,5}, FrameLabel->{"X","Y"}];
phase = StringJoin["Phase : ", (c/.fit[[1]]) 180/Pi, " Deg."];
f1=KBFComponentFrame[w1,Add->{KBFText[Text->phase]}];
TkWait[];
Exit[];
```

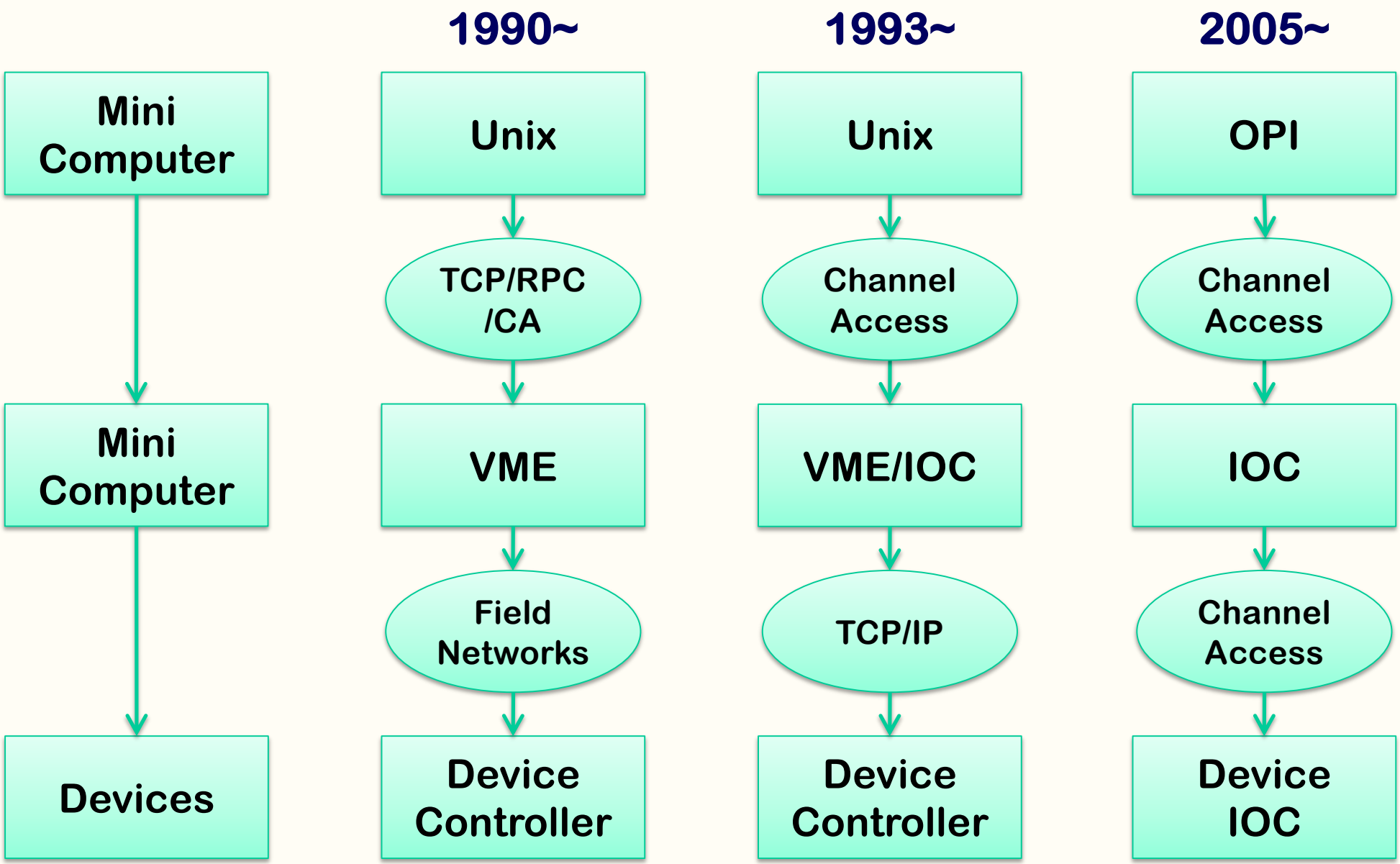
SuperKEKB での期待

- KEKB 制御のうまく行った部分を受け継ぐ
  - EPICS
  - Scripting languages
  - With simple rejuvenation of software/hardware
- さらに 2 つの追加概念

CA Everywhere

- **EPICS の通信機能 Channel Access (CA) をどこででも**
  - 装置に **EPICS** の制御機能 **(IOC)** を組み込んでしまう
  - 可能な場合には **Fieldbus/Fieldnetwork** を省略できる
  - ソフトウェアや通信機能の設計や試験の手順を大幅に削減でき、維持保守の見通しを良くする

Transition of Architecture



Overview of controls at KEK

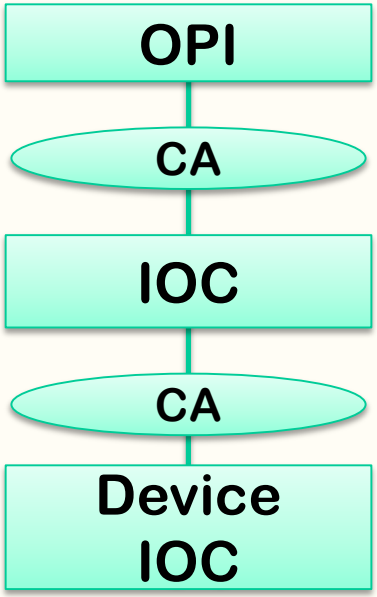
- **VME + Unix (1990~)**
  - Standard model (later EPICS) configuration
    - With several fieldbuses

- **Every controller on IP network (1993~)**
  - 2-layer physical, 3-layer in logical (Linac, J-PARC)

- **Every controller with EPICS IOC (2005~)**
  - Channel Access everywhere (CA Everywhere)
    - Good for rapid development and smooth maintenance
    - May need some consideration on network management

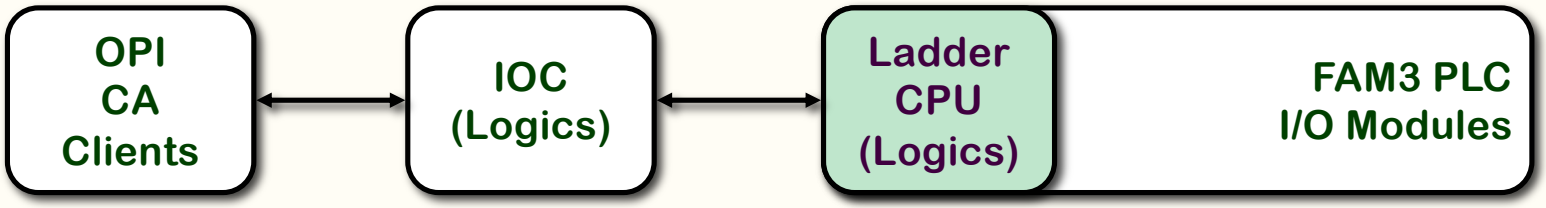
Embedded EPICS IOCs at (Super)KEKB

- Not only information server, but also the same software framework on every controller
  - Rapid development and smooth maintenance
  - μTCA LLRF module: Linux/FPGA (Odagiri…)
  - Yokogawa PLC: Linux CPU (Odagiri…)
  - Oscillo. 50Hz measurement: Windows (Satoh…)
  - MPS management :Linux/FPGA (Akiyama…)
  - Timing TDC: Linux/Arm (Kusano…)
  - Power modulator: Linux/FPGA (Kusano…)
  - Libera BPM at 50Hz: Linux/FPGA (Satoh…)
  - NI cRIO : CAS/FPGA (Odagiri…)
  - Many more…

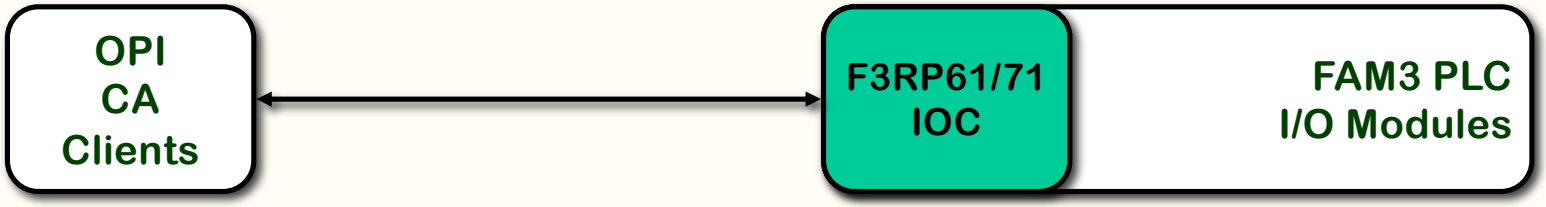


Simpler PLC Usage under EPICS

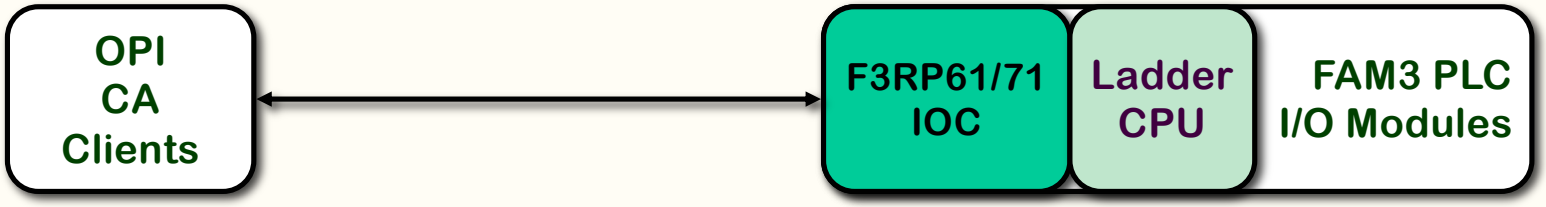
Conventional PLC usage with asynchronous access



Embedded EPICS usage with synchronous access



If necessary, we can combine



Logics are confined in PLC, and management is easier

Dual-layer Controls

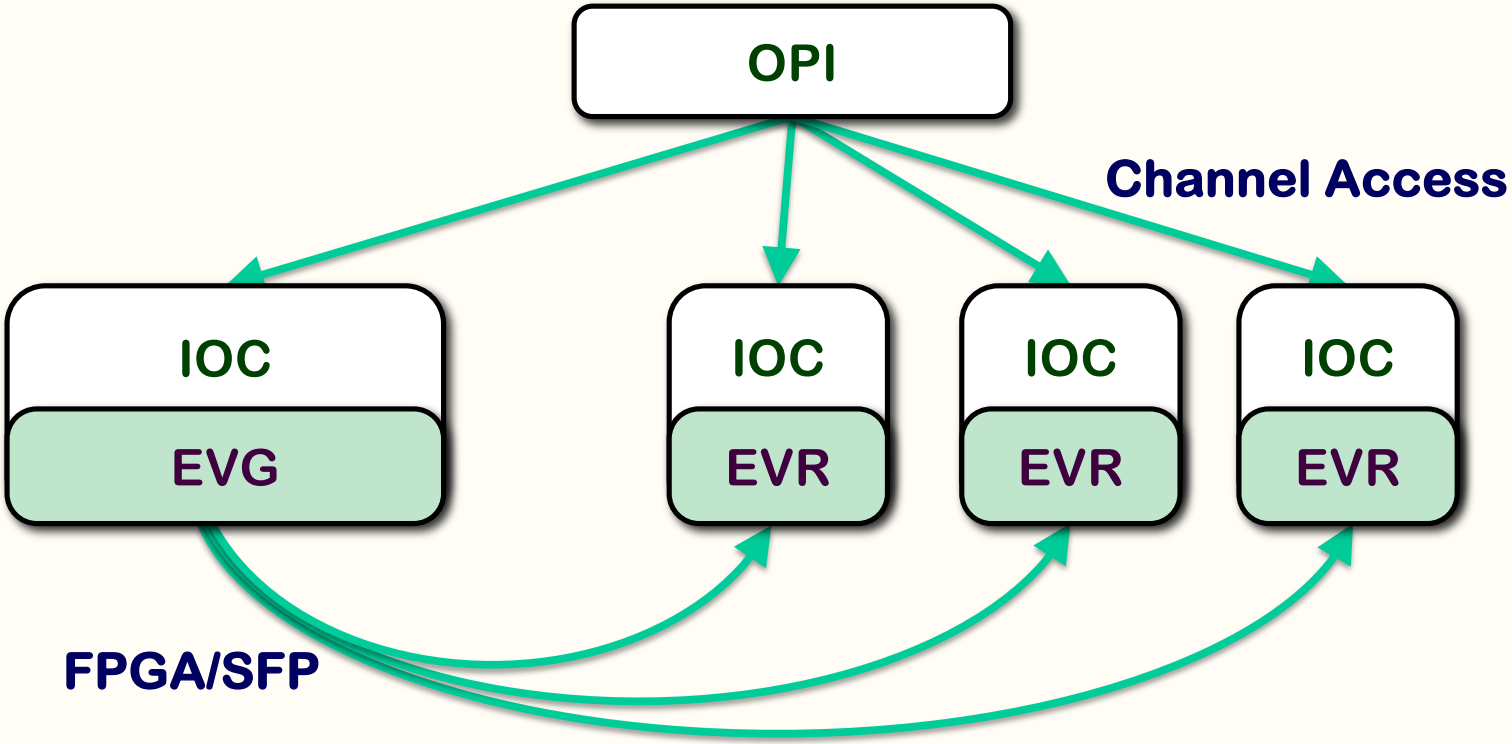
◆EPICS/CA に加えもう一つの同期制御層を追加

❖EPICS CA のミリ秒程度の同期機能に加え Event system によるピコ秒までの同期機能を可能にする

❖制御機能、同期機能と速度を追加する

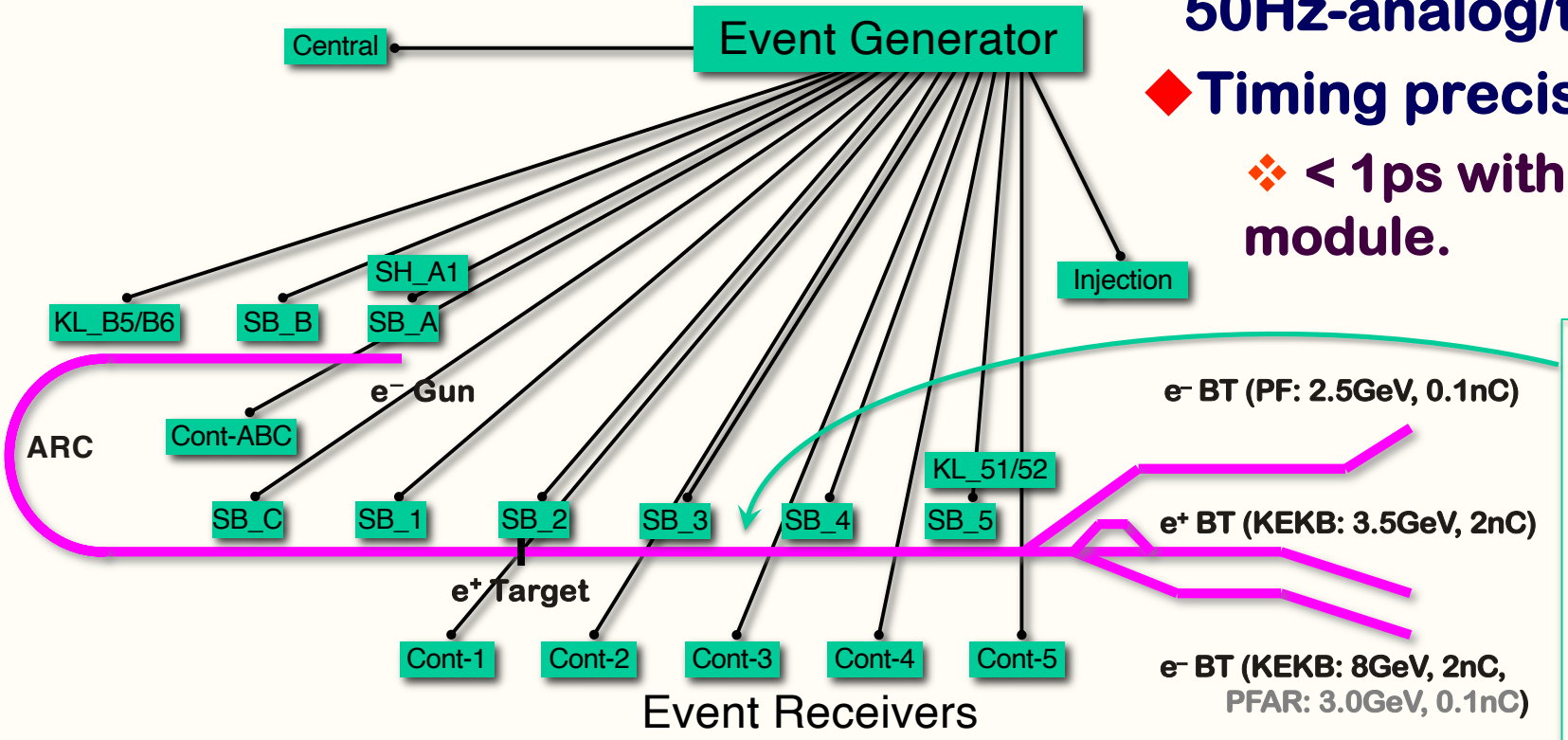
Dual-layer Controls

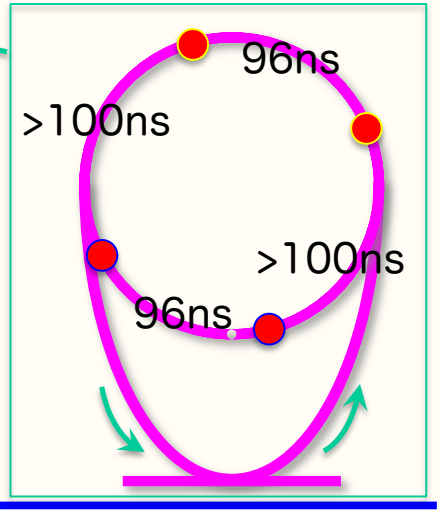
- IOC controls via Conventional EPICS CA
  - Above 1ms, ordered controls
- Fast FPGA controls via SFP/Fiber
  - 10ps ~ 100ms, 114MHz synchronous controls (KEKB)



高速広域同期制御

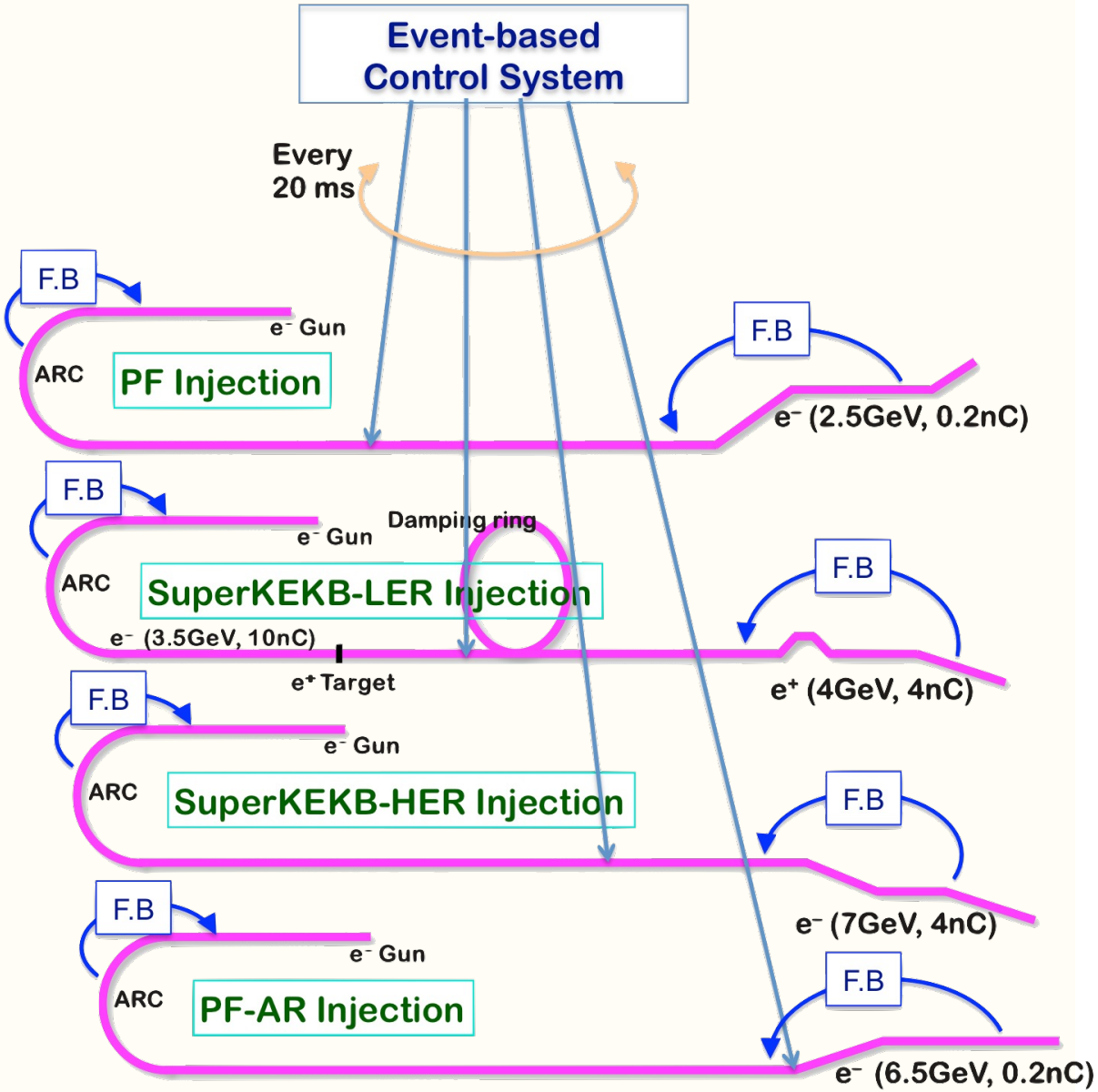
- MRF's series-230 Event Generator / Receivers
- VME64x and VxWorks v6.8
- EPICS R3.14.12 with mrfioc2 driver
- 17 event receivers in 2009, now ~100
- 114.24MHz event rate, 50Hz fiducials
- More than several hundred 50Hz-analog/timing param.
- Timing precision is < 10ps.
  - < 1ps with external module.





パルス変調運転 (PPM)

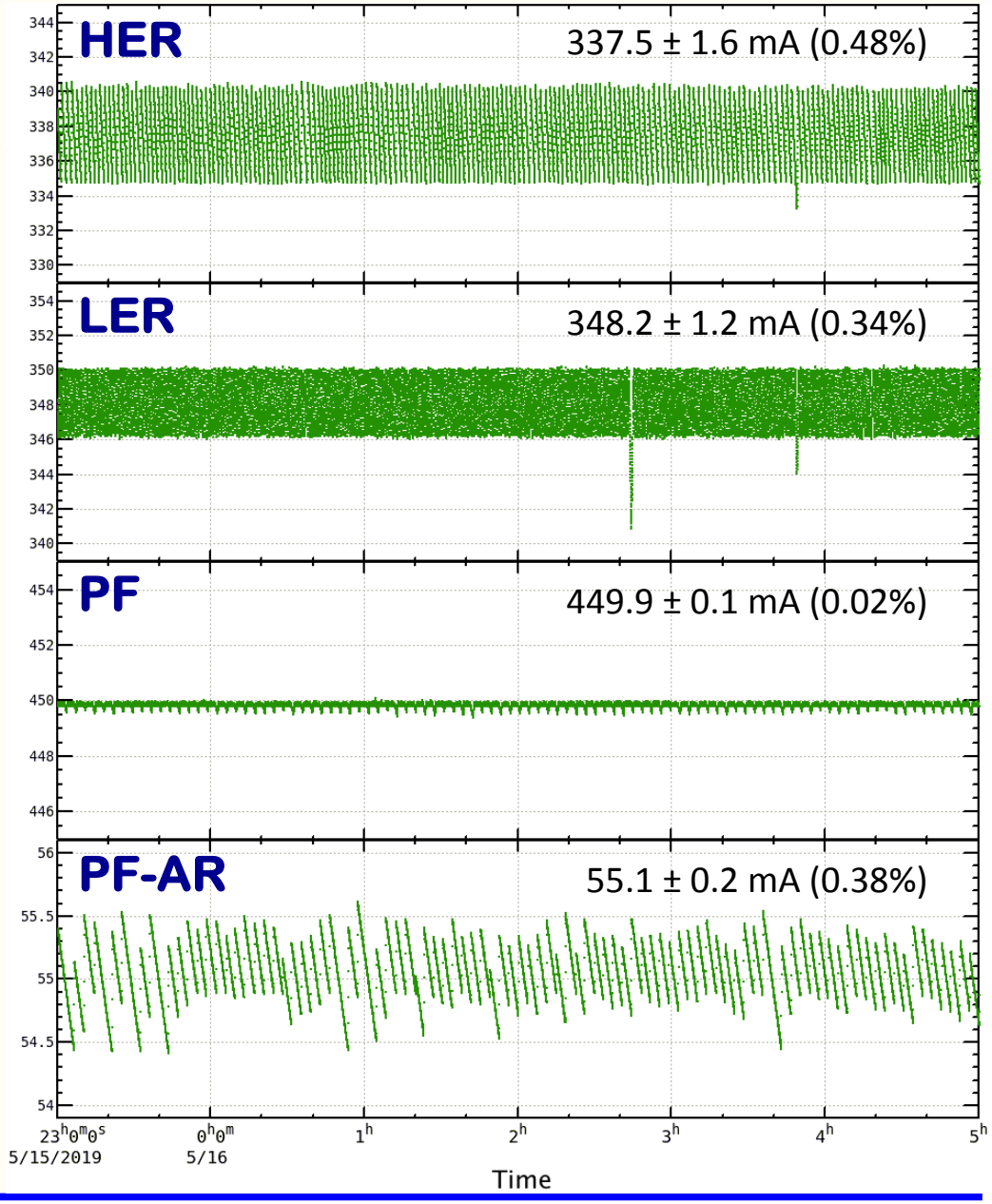
- 結果として 1 つの入射器を 4 つの仮想入射器のように振る舞わせることができる
- 20 ミリ秒毎に切り換える
  - (CERN/PS は 1980 年代から 1.2 秒毎にビームを切り換え)
- 2019 年 5 月から運用



- 仮想加速器という言葉は計算機上のシミュレーションによる加速器に使われることもある

4 + 1 蓄積リングへの同時トップアップ入射

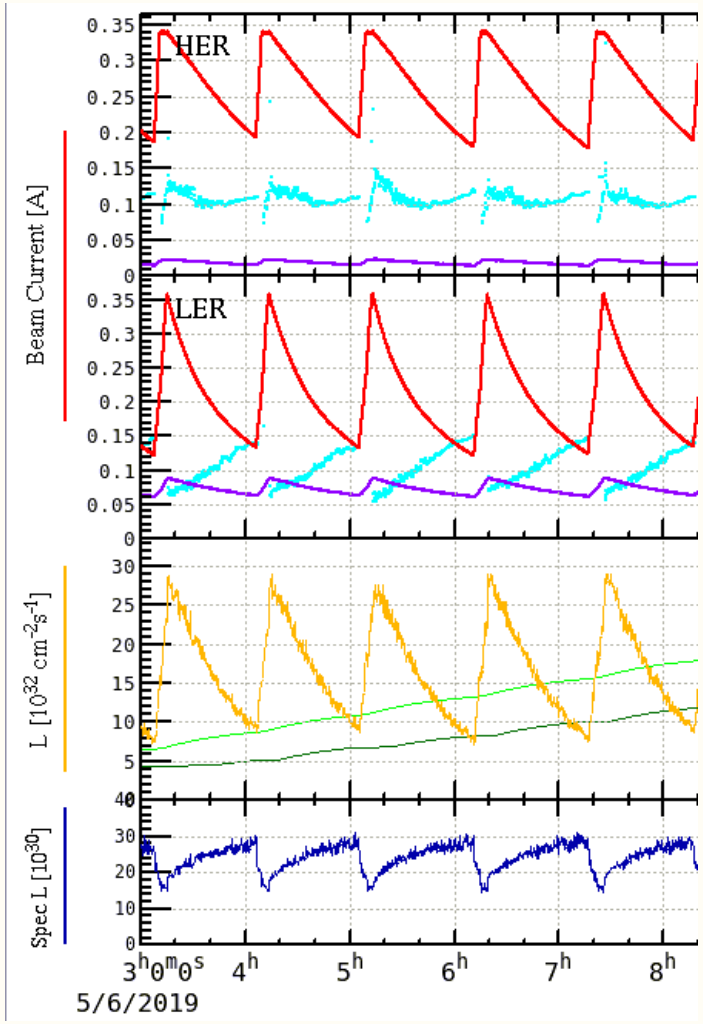
- **2019 年より予定通り運用**
  - SuperKEKB HER 7 GeV e–
  - SuperKEKB DR and LER 4 GeV e+
  - Photon Factory 2.5 GeV e–
  - PF-AR 5.0 / 6.5 GeV e–
- 4 種類のビームを 20 ミリ秒のパルス毎に切り換え
  - 200 以上のパルス装置の動作を変更
- Belle II における入射バックグラウンドの試験も成功
- 各蓄積リングの電流の平滑化を達成



2019/5/15 の
各蓄積リングの電流

SuperKEKB 同時トップアップ入射の効果

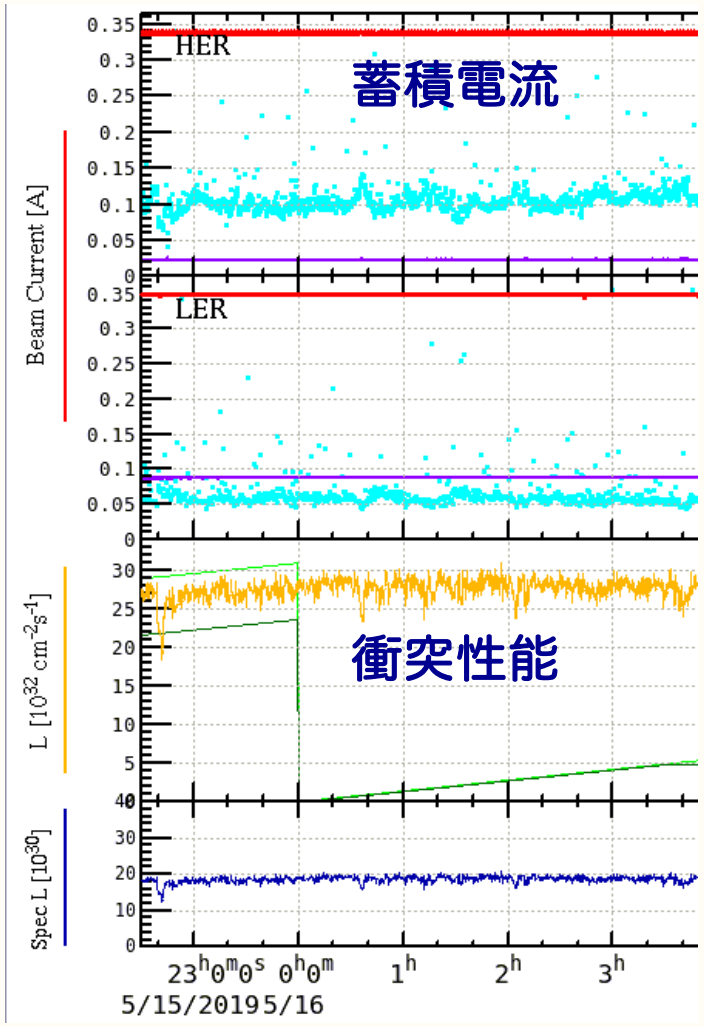
❖積分ルミノシティの向上 (好調時の一例)



17.54 /pb in 5.15 hr
(5 fills)
on May.6

41.64 /pb in 5.15 hr
(top-up)
on May.16

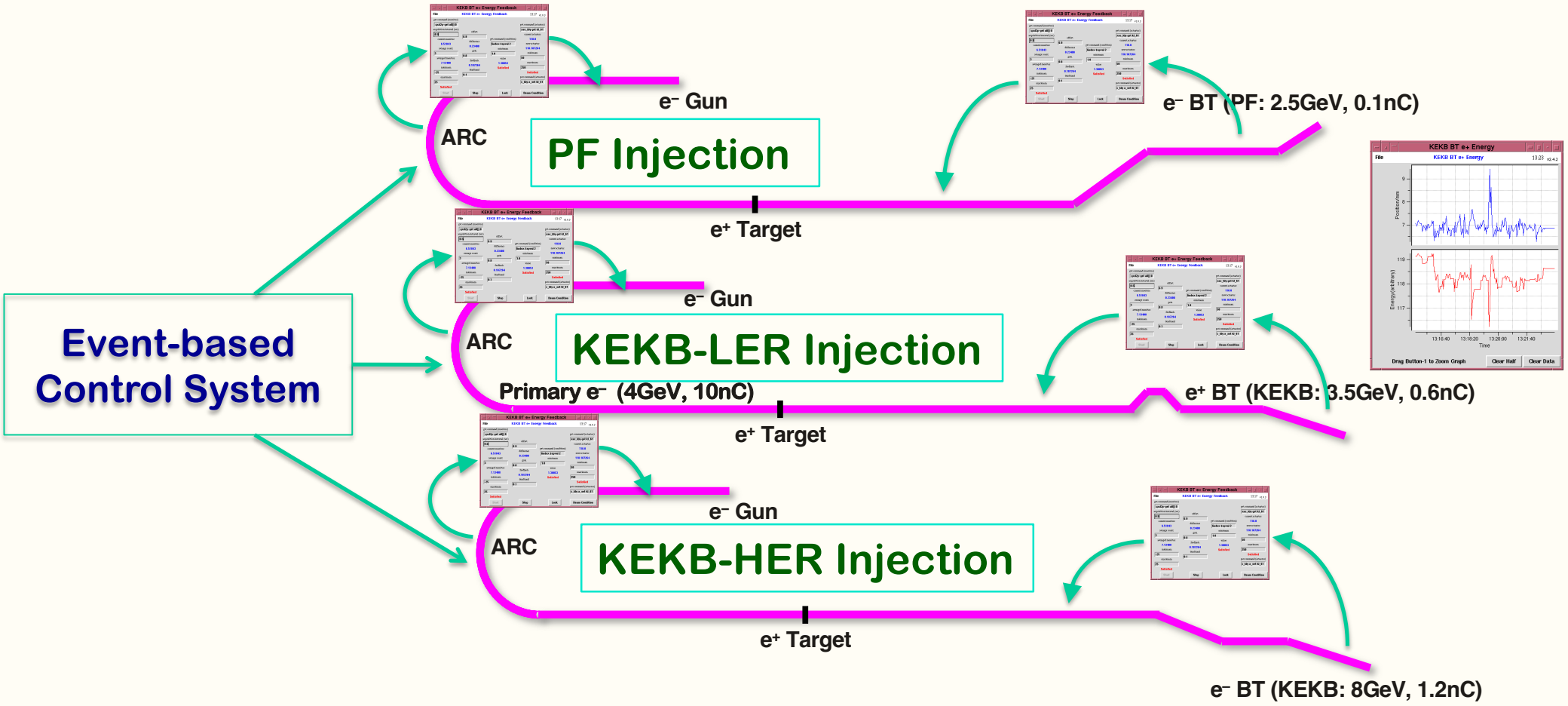
237% の
衝突性能向上



❖今年 6 月の世界最高ルミノシティの達成にも貢献

Multiple Closed Loop Controls Overlapped

- **Closed loops can be installed on each VA independently**
  - **Tested at KEKB**



Summary

まとめ

- ◆加速器制御の重要性
  - ❖全ての加速器技術と関わる
- ◆国際協力・共有できる制御
  - ❖努力と資源を無駄にしない
- ◆アイデアを成果に繋げられる制御
  - ◆ 実現する方法はたくさんあるが、各装置の能力を引き出し、性能向上のための新しいアイデアを実現し、物理実験の成果を最大限に高める方法はそれほど多くない
  - ❖性能を絞り出すことができる
  - ❖加速器の設計と現実の違いを探索
    - ⌑ 可能であれば修正
  - ❖さらに設計以上の性能を得る方法を提示
    - ◆ 今後は特に **Application/Operation Software** も共有できる可能性が高く、同じ基盤の上で **Collaboration** のアイデアを交換できるかも

    - ◆ http://www-linac.kek.jp/linac/
    - ◆ http://www-linac.kek.jp/cont/epics/

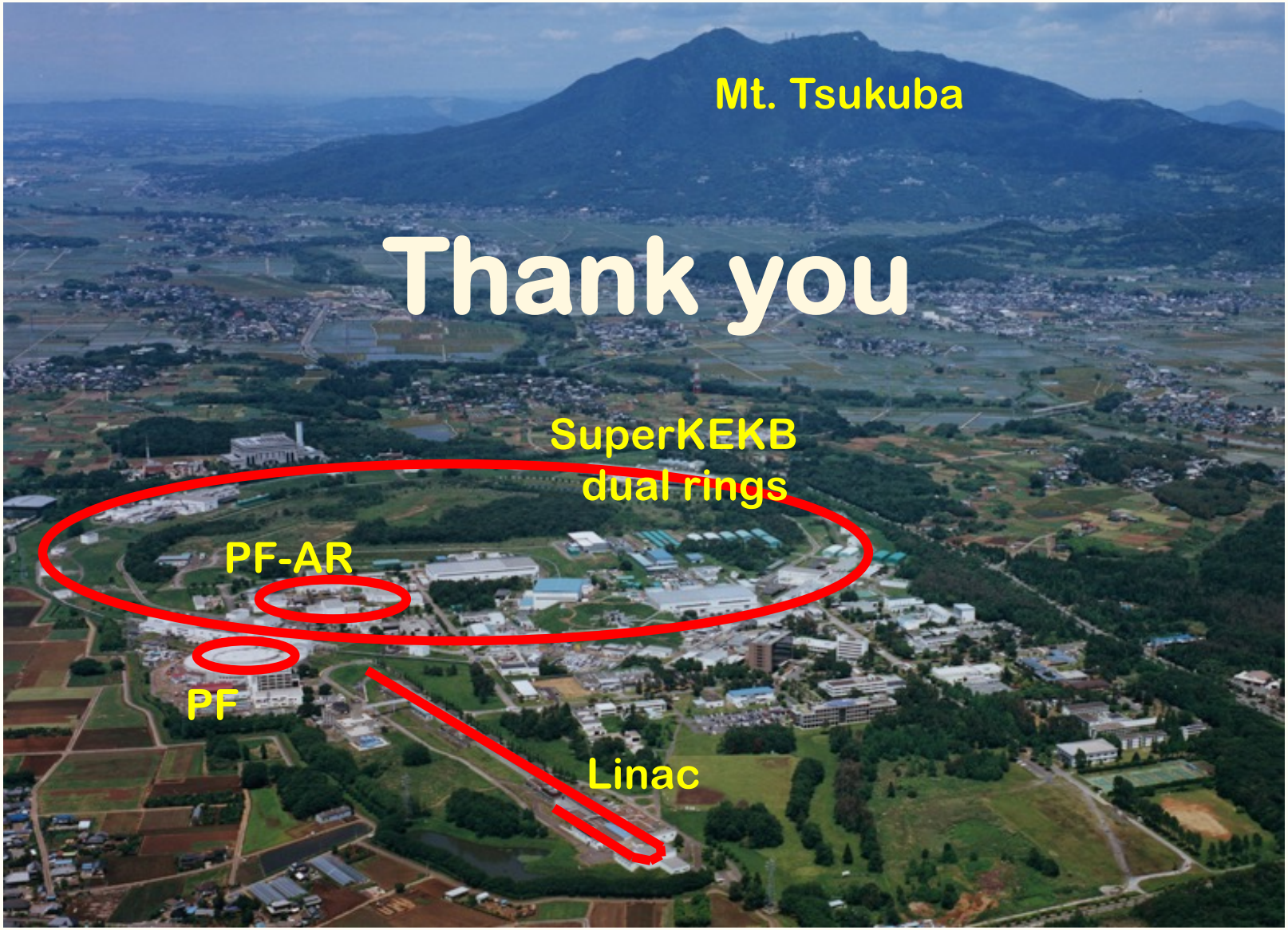
Phronesis

◆アリストテレスの知恵の考え方

- ❖**Sophia** とは逆に普遍的な真実を理解しようとする能力(らしい)、全体的な解決方法を探す能力

◆加速器制御

- ❖実現する方法はたくさんあるが、各装置の能力を引き出し、性能向上のための新しいアイデアを実現し、物理実験の成果を最大限に高める方法はそれほど多くない
- ❖同じ基盤の上で各グループがアイデアを交換できることが重要なのではないか


Mt. Tsukuba
Thank you
SuperKEKB
dual rings
PF-AR
PF
Linac

Thank you