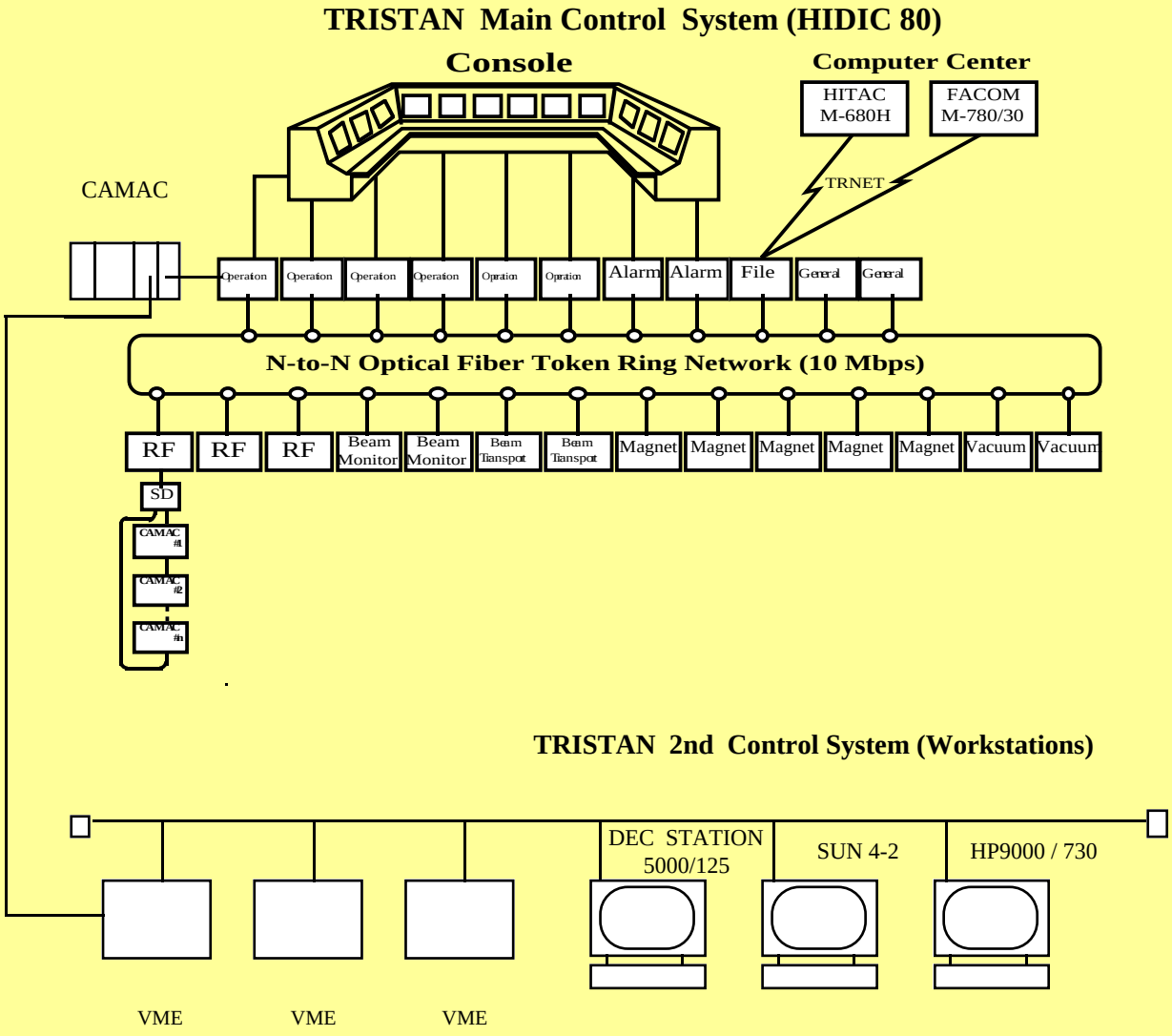
Accelerator Model in an Accelerator Control System

--- the Case of TRISTAN and KEKB ---

Noboru Yamamoto
KEK, Accelerator Facility

TRISTAN Control System



- Modeling Program runs on
  - HIDIC Mini computers : in-house optics calculation program written in NODAL.
  - Mainframe computers : PETROK, MAGIC
- Model Programs are used for:
  - Orbit Correction/Control
  - Optics control
    - Twiss parameters
    - Tune
  - Chromaticity correction
- Data Transfer Method: Files

Problems in TRISTAN Control System

- Several Accelerator models in various applications.
  - Discrepancy between models
  - Different input format
  - Difficult to maintain consistency between models.

SAD

- Developed in KEK for Accelerator Simulation since 1986.
- 1-fits-All for Accelerator Study
- SAD Script Programming Interface in *Mathematica* Style

- Channel Access Interface
- Python/Tk Interface

SAD:1-fits-All for Accelerator Studies

- Structural Definitions of Beam Line & Component
  - Elements
    - DRIFT
    - BEND,QUAD,SEXT,OCT,DECA,DODECA,MULT,ST,SOL
    - CAVI
    - MON, MARK, APERT
  - Beam Line(LINE) : List of Elements and/or Lines
- Optics Matching
  - Optical/Geometrical matching
  - Off-momentum matching
  - Finite-amplitude matching
  - Spin Matching

SAD:1-fits-All for Accelerator Studies (cont'd)

- Particle Tracking
  - 6D full-symplectic tracking
  - Dynamic aperture survey
  - Synchrotron radiation
- Nonlinear Analysis
  - Taylor map by automatic differentiation
  - Lie algebraic map
- Emittance Calculation
  - 6D Beam-matrix method
  - Anomalous emittance
- Spin de-polarization

SAD:Script Programming Interface in *Mathematica* Style

- Built-in, System- and user-defined functions for accelerators
  - Data types
    - (Real) Number, String, List
  - Functions for Beam Optics
    - Twiss, BeamMatrix, OptimizeOptics, ...
  - Flow control
    - IF, Do, For, ...
  - Input/Output
  - Graphics
  - Numerical Functions
    - Sin,Cos, BesselJ,....
  - List Manipulations
    - First, Last, Join, Append, ...
  - Functional Operations
    - Apply, Map, Thread,....

SAD/FFS functions:

Constants:
Degree GoldenRatio I INF* Infinity NaN* P
SpeedOfLight

Elementary-functions:
ArcCos ArcCosh ArcSin ArcSinh ArcTan
ArcTanh Cos Cosh Exp Log Sin Sinh Sqrt Tan
Tanh

Special-functions:
BesselI BesselJ BesselK BesselY Erf Erfc
Factorial Gamma LogGamma LogGamma1
GammaRegularized GammaRegularizedQ*
GammaRegularizedP*

Numerical-functions:
Abs Ceiling Floor Max Min Mod Round Sign

Matrix-operations:
Det Eigensystem IdentityMatrix Inner
LinearSolve Outer SingularValues Transpose

Random-number:
GaussRandom* Random* SeedRandom

Complex:
Complex ComplexQ Conjugate Im Re

Fourier-Transformation:
Fourier InverseFourier

Data-Manupilation:
FindRoot Fit*

Minimization:
DownhillSimplex*

List-manipulations:
Append Complement Delete Depth Dimensions
Drop Extract Flatten FlattenAt HeldPart Insert
Intersection Join Length Part Partition Prepend
Product Range ReplacePart Rest Reverse Select
Sort Sum Take Table Union

Character-strings:
FromCharacterCode CharacterPosition
StringDrop StringInsert StringLength
StringPosition Symbol SymbolName ToCharacteCode
ToLowerCase ToUpperCase ToExpression

Functional-Operations:
Apply Cases Count DeleteCases Identity Fold
Function Level Map MapAll MapAt MapIndexed
MapThread Nest Position Scan SelectCases*
SwitchCases* Thread

Flow-Control:
Break Catch Check Continue Do For Goto If
Label Return Switch Throw Which While

Tests:
AtomQ ComplexQ MatchQ MatrixQ MemberQ
NumberQ Order VectorQ

Input/Output:
Close Flush* Get OpenRead OpenWrite OpenAppend Print Read Write WriteString

Scoping:
Block Module With*

Attributes:
Clear Evaluate Head Hold Protect ReleaseHol SetAttributes* Unevaluated Unprotect

Graphics:
ColumnPlot ListPlot Plot Show OpticsPlot FitPlot

System Interface:
Directory Evironment Fork GetEnv* GetGID* GetPID* GetUID* SetDirectory System* TemporaryName* Wait

Utility:
Date DateString Definition FromDate ToDate Pause MemoryCheck* Message Off On Sleep TimeUsed Timing TracePrint

Functions listed above work basically in the same way as Mathematica's except those marked by *.

FFS-dedicated-functions:
BeamMatrix CalculateOptics DynamicApertureSurvey Element Emittance FI FitValue FitWeight LINE OptimizeOptics RadiationField RadiationSpectrum SymplecticJ SetElement TrackParticles Twis VariableRange

Beam-line-functions:
BeamLine BeamLineName ExtractBeamLine PrintBeamLine WriteBeamLine

SAD for Control system

- Control Algorithm is developed in SAD and test in simulation on SAD.
- It provides natural concept, like Twiss parameter, for an accelerator physicist.
- Accelerator physicist controls an accelerator model using SAD script -> Why not controllers an accelerator using SAD script.

SAD:Channel Access Interface

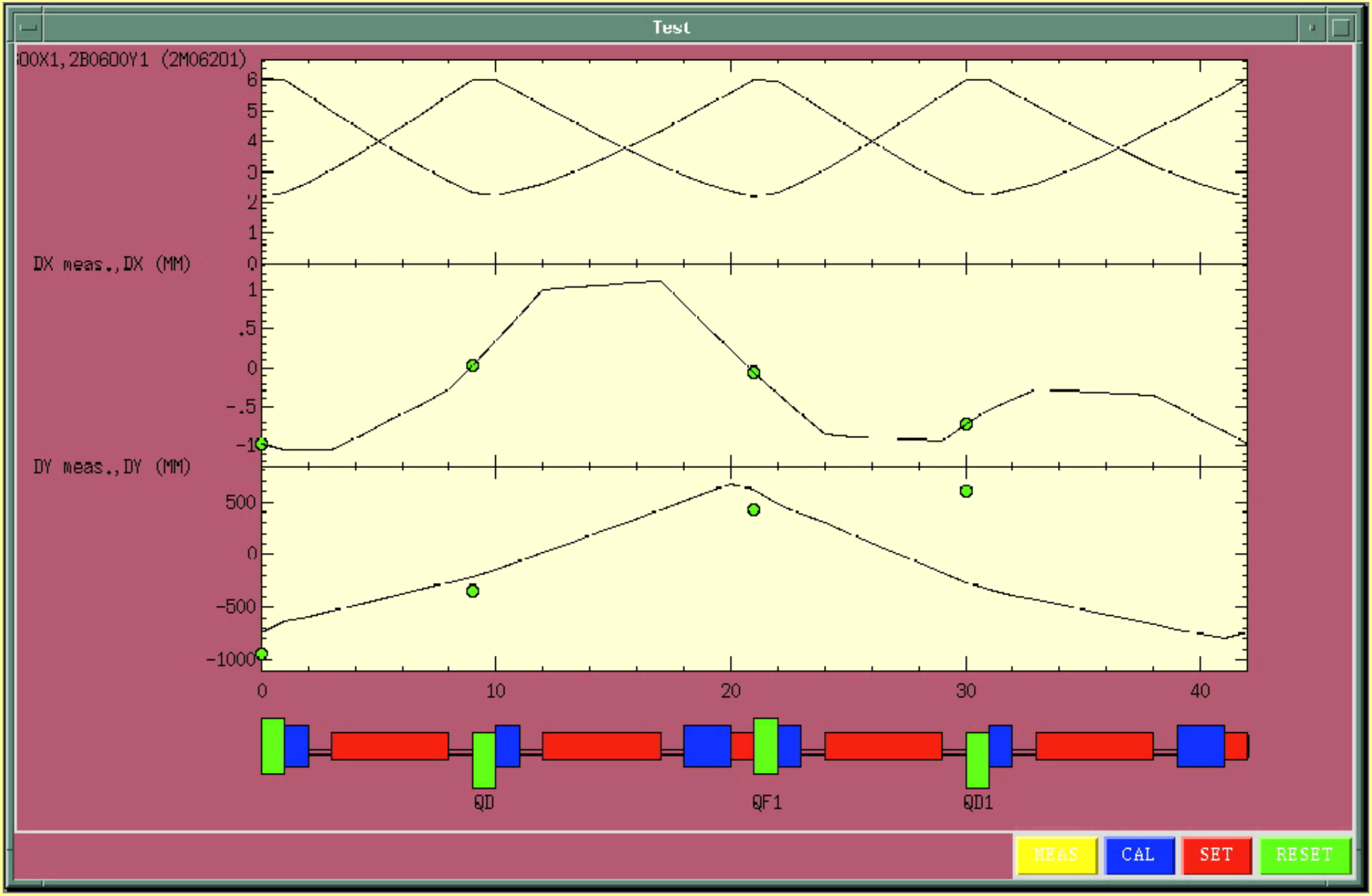
- Four Functions for CA.
  - CaOpen[]
  - CaRead[]
  - CaWrite[]
  - CaClose[]

- Argument for these functions are:
  - String (channel name)
  - Number(channel ID returned by CaOpen)
  - A List of these
    - bpms=CaOpen[{“BPM1”,”BPM2”,....}]
    - orbit=CaRead[bpms]

SAD:Python/Tk interface

- Graphical user interface is required for the modern control system
- Tk library provides a simple framework to develop GUI.
- SAD can access Tk widget through Python/Tk library.
  - Text
  - Graph(Canvas, BLT)
  - Slider
  - Button.....
- Python - Oracle interface is being tested.
  -> ORACLE database access from SAD

A Sample display : SAD/Tkinter.



SAD

- provides a rich set of functions to control or to analyze an accelerator.
- can interact with the control system through CA.
- can interact with an operator through GUI.