

LLRF05

10.10.2005

Automation of large scale



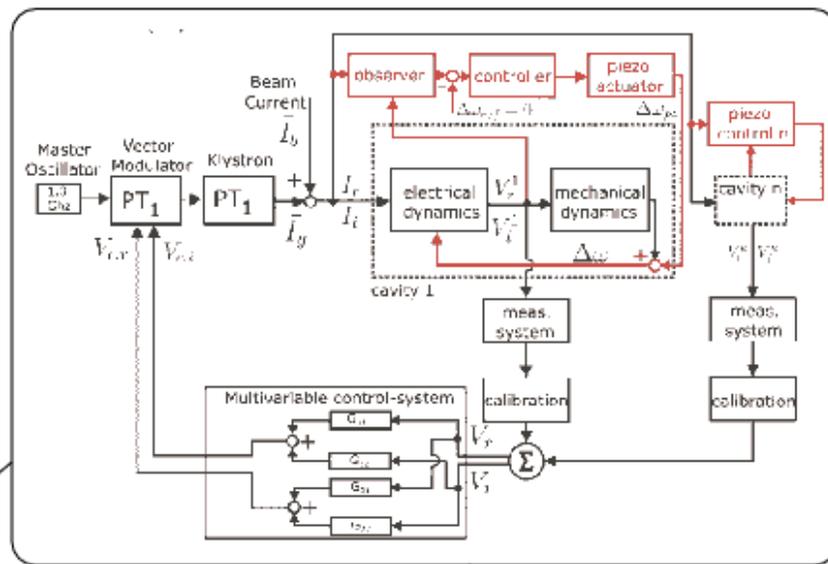
RF systems

Markus Hoffmann



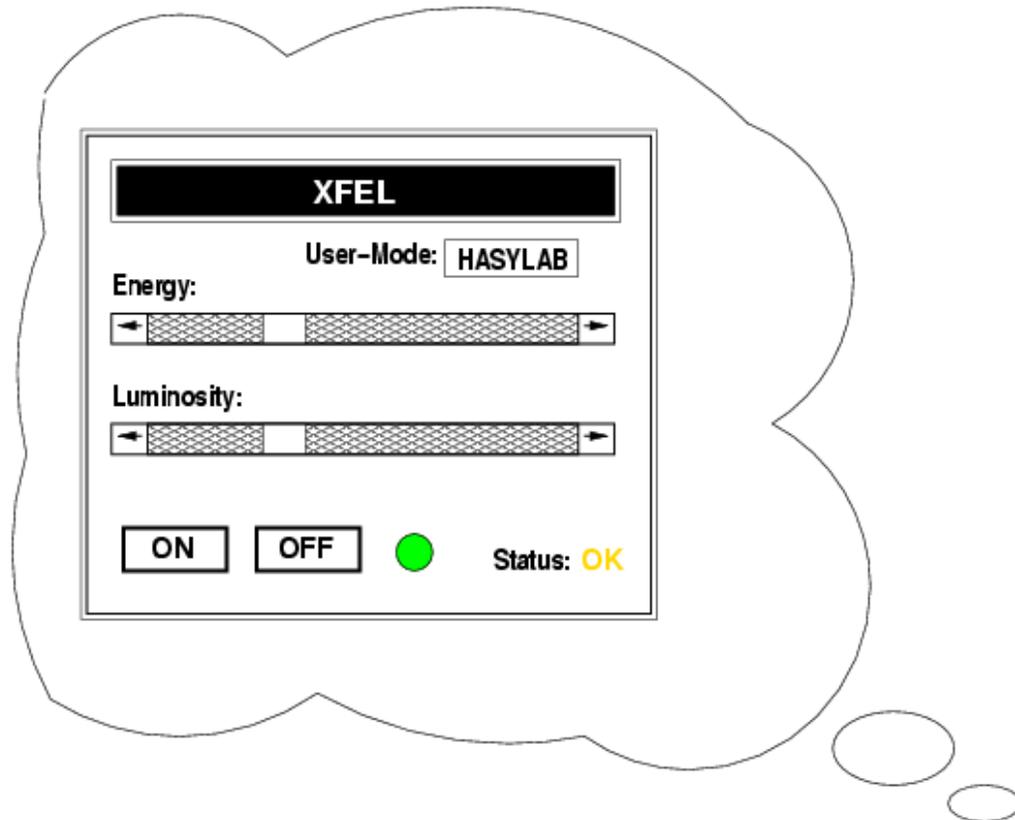
- **concept framework components**
- **state machine**
- **NOT: realization**

RF-System



XFEL, ILC

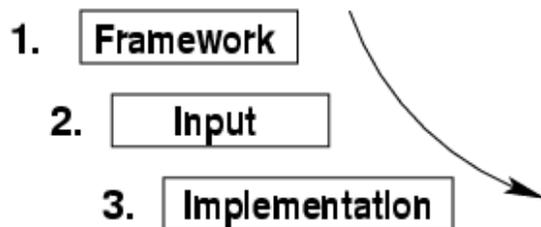
User-front-end



... its a dream ...

AUTOMATION

- high complexity
- up-time and reliability as high as possible
- scalable
- transparent to experts
- compatible, use existing infrastructure
- flexible, open to extensions
- Operation (human) as simple as possible
- foolproof (robust against human failure)

1. Framework
 2. Input
 3. Implementation
- 

Needs:

- good and robust concept
 - map reality to control system (observables)
 - integrate "all" expert knowledge to system
 - define standards and interfaces (logical structures)
 - do not allow deviations from concept
- (even if realization of subsystem would be more easy or simple)

modular

Avoid Rank Growth !

comments:

rank growth is a "good" concept. It works and many existing accelerators like DESY-HERA-PETRA-TTF use it. It is scalable but a full automation and exception handling is actually impossible.

Istead a bunch of human operators is needed to keep the machine running and for otimization of all parameters and for error detection.

Also a large pool of human experts is necessary to track exeptional conditions.



Operator error is one of the most frequent sources for reduced machine availability.

This is the best way we have at the moment.

But **can we do better ?**

*** pushing to the performance limit ***

our Ansatz

for automation

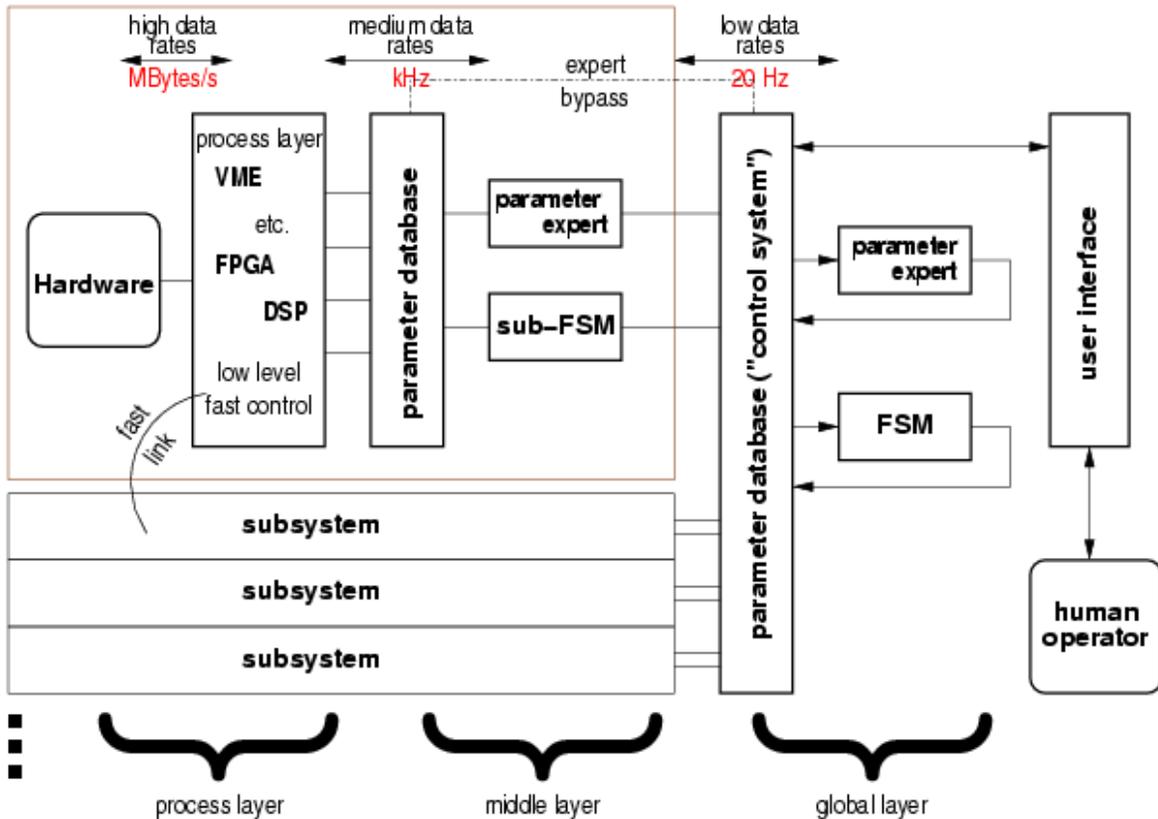
modular concept:

- 3 layers**
- 2 main components**
- 1 backbone**

minimal set of components

to operate a complex system

hardware specific hardware independent



Components (Applications) for automation

- **Parameter database** (distributed, server-client, "control system" DOOCS)
backbone
 - **User-Interface** *non intelligent*
 - **"Parameter Experts"** } ← *highly standardized*
 - **"state machines"** }
 - **individually designed** ← *less standardized*
**process layer applications
and hardware interfaces**
- expert knowlege** → TRANSFER → (to "Parameter Experts" and "state machines")

Parameter and Parameter-Expert

6

database

user/operator
action parameters

hardware parameters

calculated parameters

Parameter
Expert
routine

Trigger
by change in input value

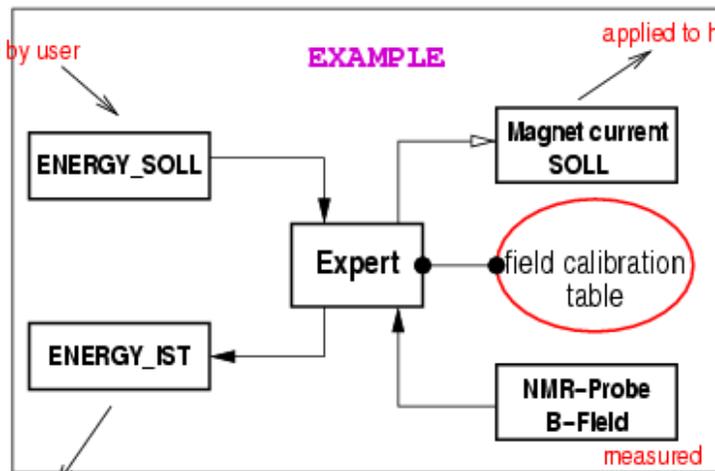
Parameter:

Name	ENERGY_SOLL
Unit	MeV
Range	[0; 1000000]
Resolution	1e-2
Type	float
Dimension	0
Size	1
Value	920000.0

changed by user

EXAMPLE

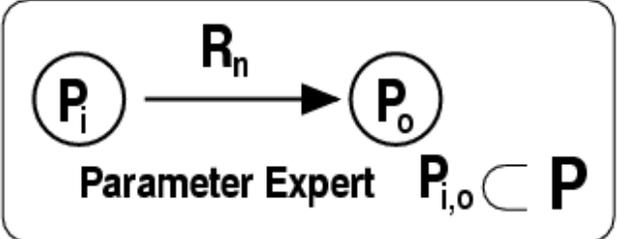
applied to hardware



"Zero-time" calculation !

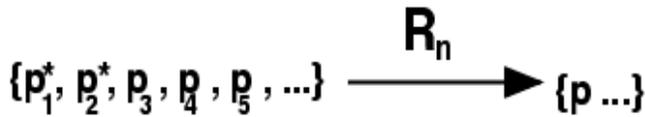
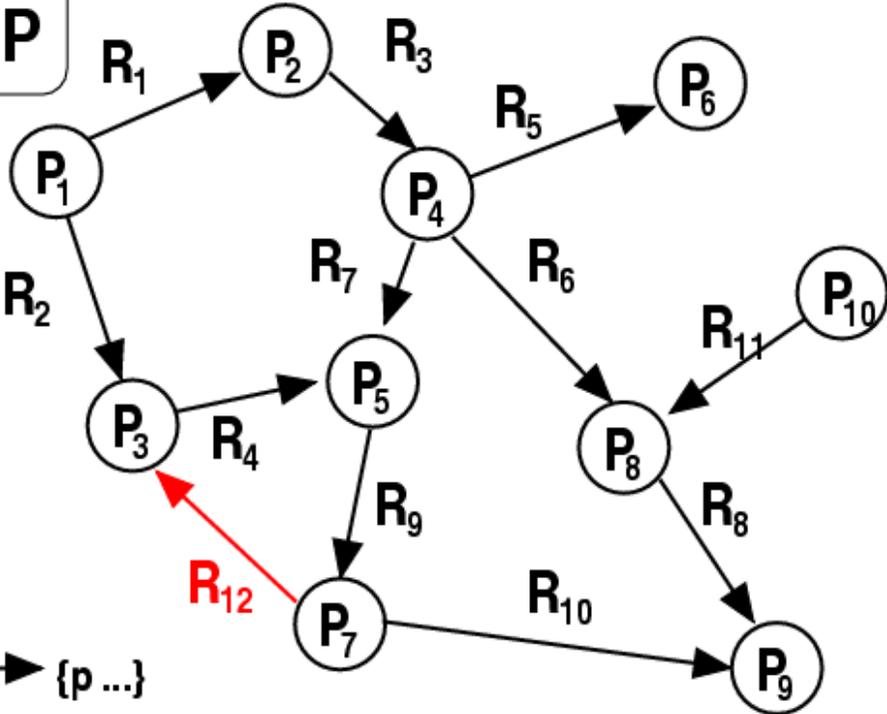
backbone

displayed to user



"rule" machine

Cycles:
 allowed but careful!
 → Resolution !



quantity of parameters and trigger parameters

transformation

Directed Graph

States and State-Machines

7

FSM = Finite State Machine

digital sequential logic

operates based on current status information and the current **state**

Outputs are generated in response to the inputs and current state

State ON, OFF, ERROR, Operation Mode A, ...
unique identifier for possible current situations

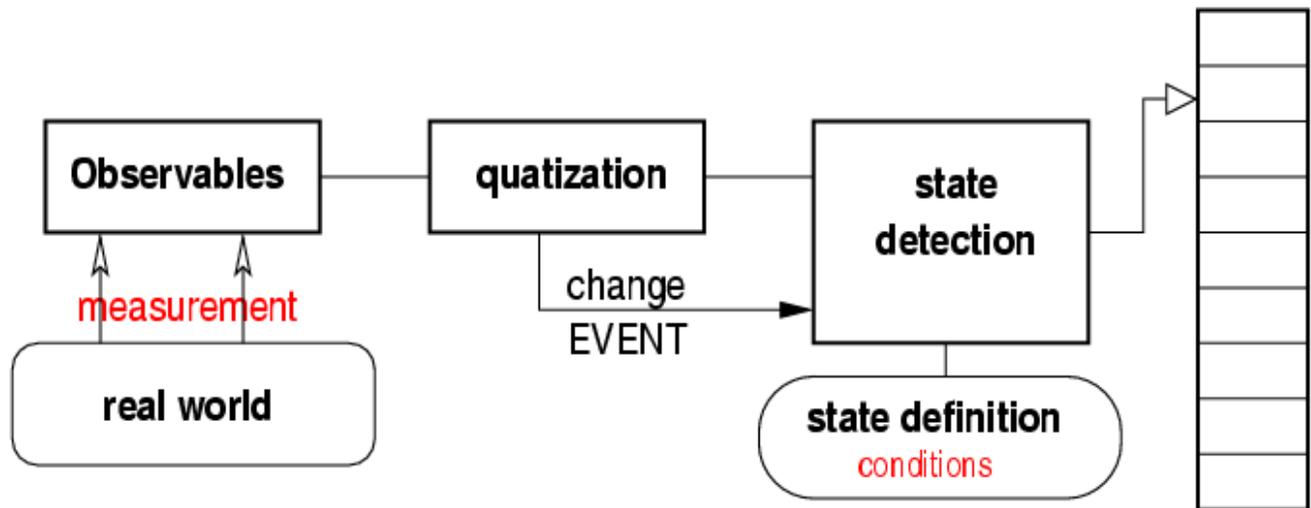
Transition from OFF to ON
a path to be taken by a state machine from state to state

Routine (process, sequence, flow) perform switching actions
performs changes to outputs, also sequences

State definition

- quantization of observables
- mapping of reality to a **finite set of states**

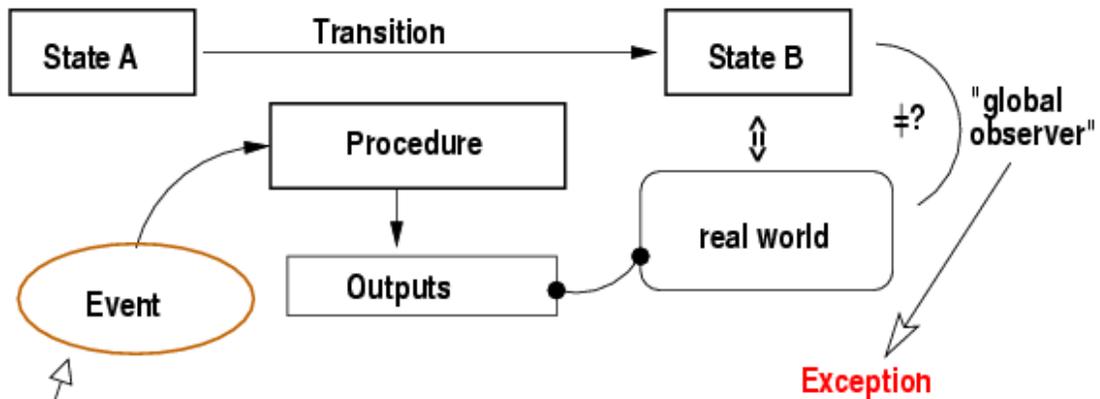
which have to be defined in a smart way to represent a state of a complex system



There can be a global set of states and also subsets of sub-states, but each level of states must be a partition of the whole state space the system can ever be.

States and Transitions

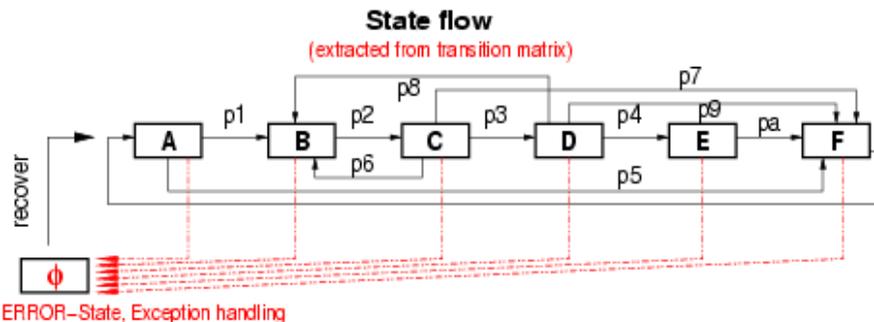
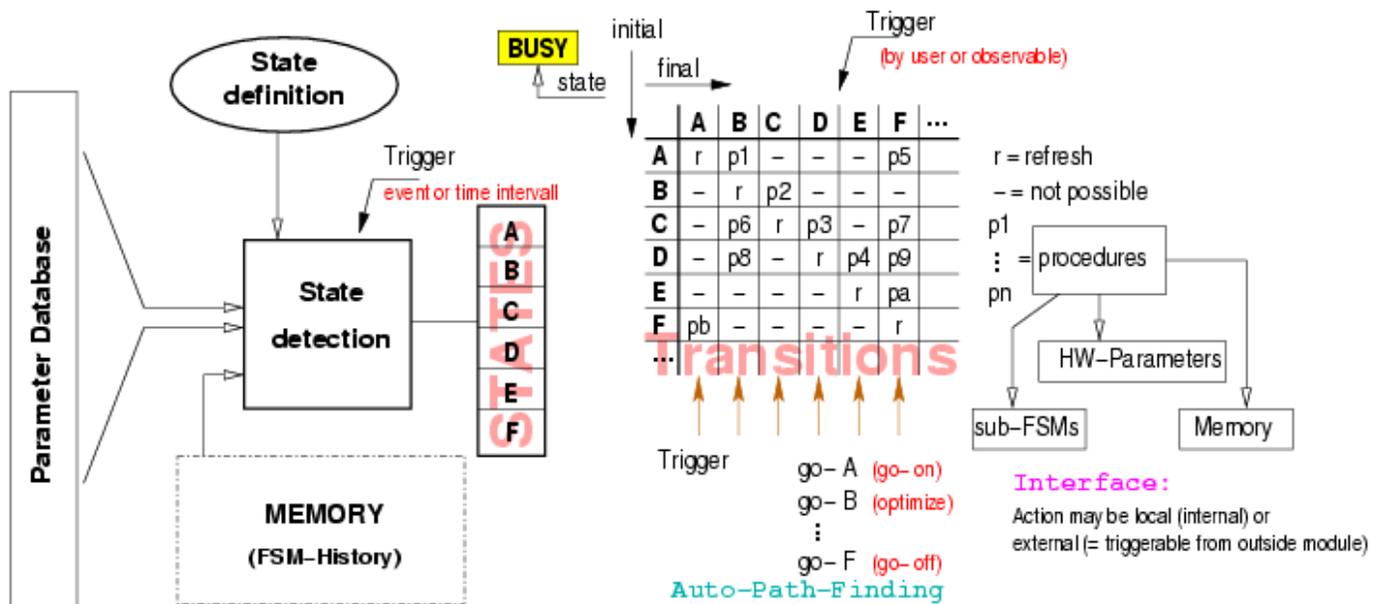
- Operation of a System can be described in Transitions from one state to another either triggered by user or by a condition on inputs (observables) and exceptions of the state machine **EVENTS**
- Each Transition is accompanied by one ore more procedures which reflect the expert knowledge how to operate the system in a resonable way **Actions**



if no trigger, nothing (which would change state) should happen !

NO TRANSITION WITHOUT TRIGGER (eg. by change in Inputs) !

Module of FSM



Interpretation:

- A = ready
- B = STBY
- C = on, stable
- D = on, optimizing
- E = shutdown
- F = OFF

EXAMPLE

Tasks of "parameter experts":

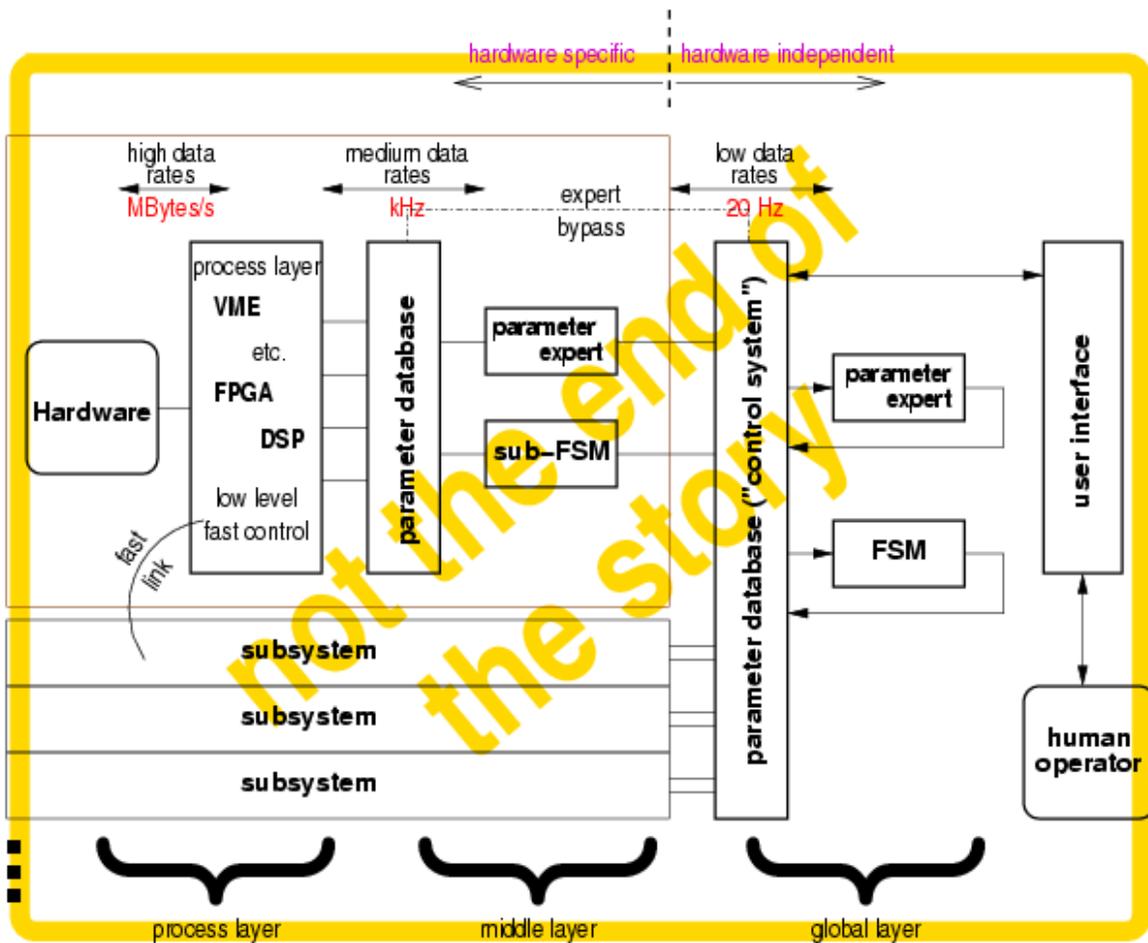
- a) allow a level of abstraction (virtual observables)
- b) allow a hardware independent "physical" view to the machine

Tasks of "state machines":

- a) normal operational procedures
- b) exception handling (Error diagnostics and procedures to recover from Error-States)

other conceptual issues:

- user interface
- integrity of observables
- parameter name conventions
(setpoint- and actual value parameters)
- transparency
- common states for all systems (conventions)
- tools
- evolution process (extensions, improvements)



conclusion:

- no full automation realized at any accelerator so far
- ideas for concepts are there
- layers, database, parameter experts, state machines, user interface
- Quality of Observables !
- Systematic evolution must be part of the concept
- need to talk about "tools" for implementation
- needs discipline to freeze the starting concept and do the implementation without rank growth

Evolution Process

