Automation for LLRF in ATCA Architecture

Bogusław KOSĘDA
Department of microelectronics and Computer Science
Technical University of Lodz, POLAND
Dec. 3, 2007

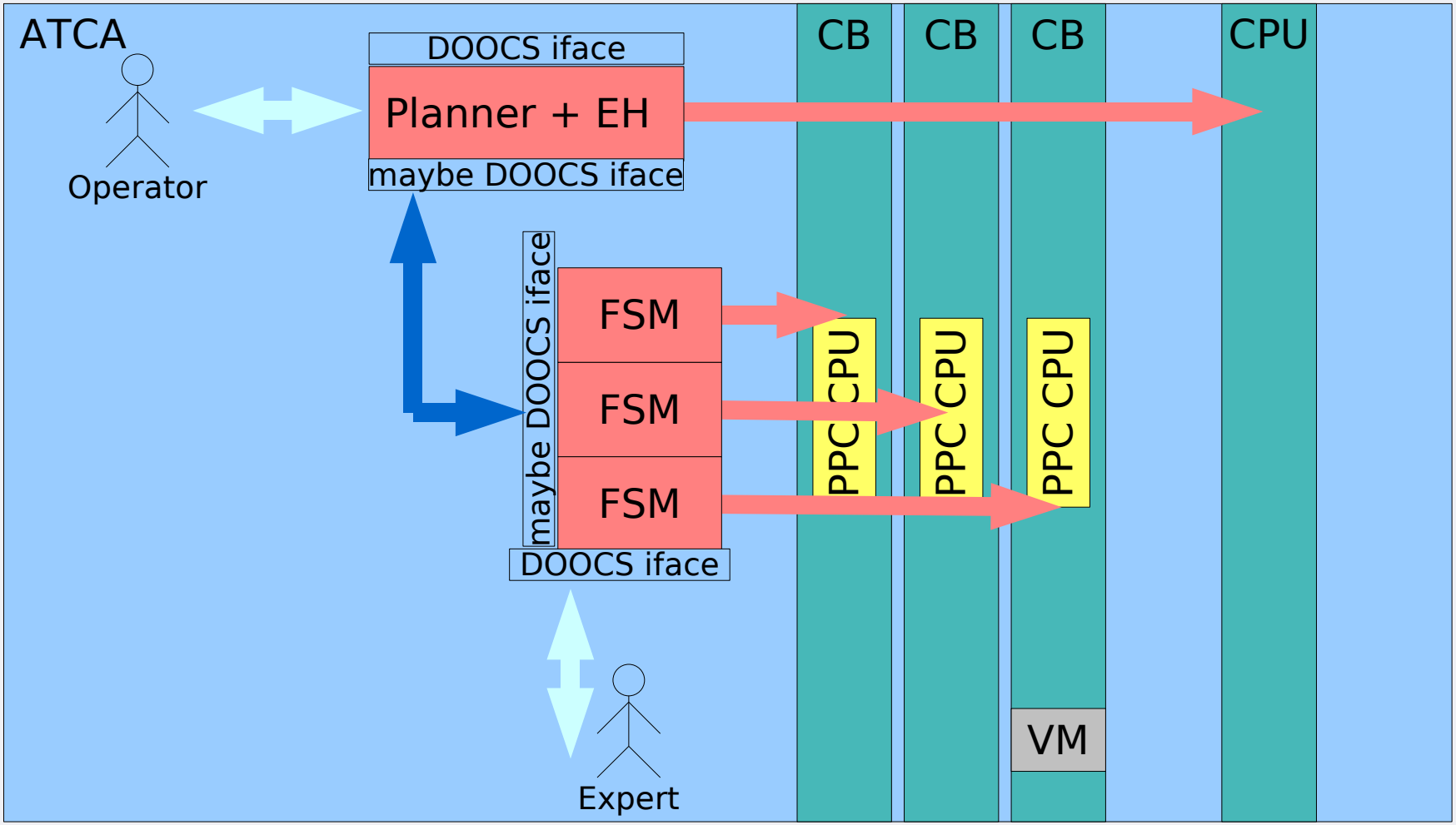
Agenda

- Tasks of automation
- General scheme of the automation in the ATCA
- Automation in the ATCA - dependencies
- Specification and implementation
- Experience
- To do
- People involved

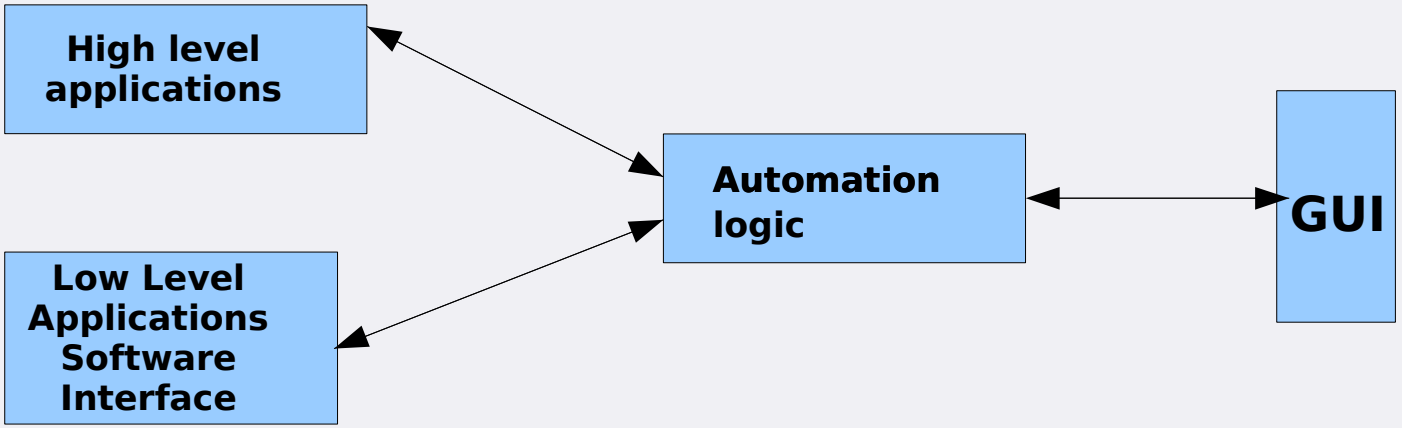
Tasks of automation

- To automate routine operators tasks
- To adapt to unpredictably changing conditions of underlying hardware
- To recover the subsystem from known faults
- To advice the operators how to proceed in case of unrecoverable breakdowns
- To perform above activities **safely!**

General scheme of the automation in ATCA



Automation in the ATCA - design



- Does not communicate directly with the hardware, so the design of the automation logic has nothing in common with the ATCA architecture.

- Situation is different for the implementation of the senses and hands of the FSM (high/low) level applications and monitoring signals, the ATCA can facilitate implementation of these applications by its performance and interconnectivity.

Automation in the ATCA

Automation relies on dozens of high/low level applications and diagnostic signals so:

- Interconnectivity of high and low level applications will be surely better supported by the ATCA than by the VME
- Better performance of the ATCA crate will better suit high network traffic generated by the automation software
- ATCA shelf manager will provide many diagnostic information used by the fault recovery automation module

Automation in the ATCA - design/spec

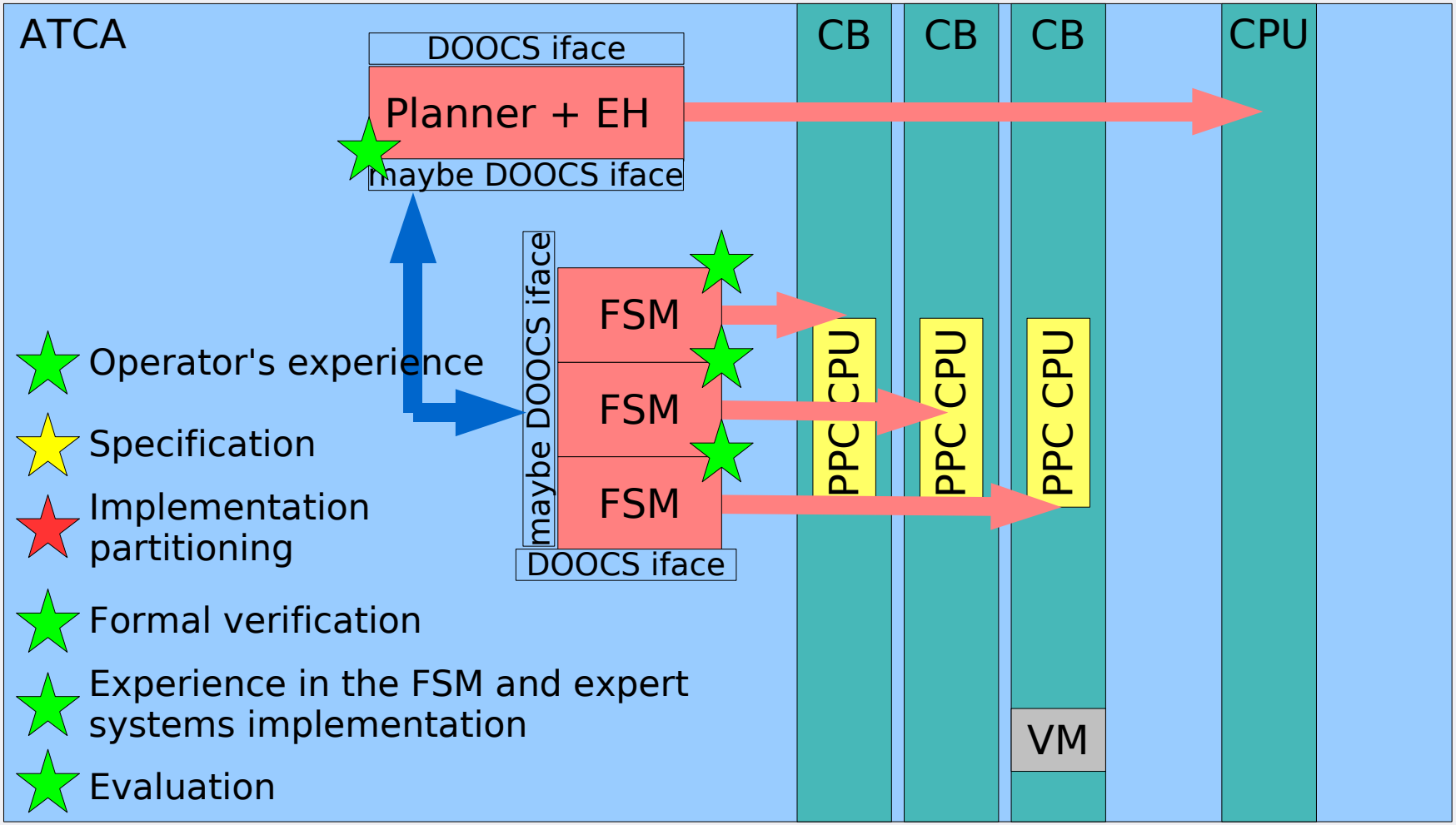
Specification – the rhapsody way:

- Use cases
- Activity charts
- Tests of the charts within the confines of operation procedures.
- Statemachines – when necessary
- Planner specification when necessary

Automation in the ATCA - implementation

- General framework
  - It would be desired to have one framework to test and verify the statemachines impemented possibly in FPGA, on-site controllers and slow supervision FSMs
- Reusability of code – hardly possible (communication protocols can be standardized)
- Extendability – difficult in the case of the FSM (distributed, not distributed is overwhelming)
- Bypassing the automation – full on, full off only – no other tricks

Experience


ATCA
DOOCS iface
Planner + EH
maybe DOOCS iface
maybe DOOCS iface
FSM
FSM
FSM
DOOCS iface
CB
CB
CB
CPU
PPC CPU
PPC CPU
PPC CPU
VM
Operator's experience
Specification
Implementation partitioning
Formal verification
Experience in the FSM and expert systems implementation
Evaluation

To do

- Specification of the LLRF system as a family of simple FSMs - they will limit system users to proper operation.
- Specification the tasks for the planner and the exceptional situations – they will implement routine operation tasks and perform fault recovery
- Remote GUI for the automation module for automating routine operator's procedures (so far. the DDD does not provide creation of dynamical GUIs)
- Optimization of the planning algorithm used by the automation.

Automation prerequisites

- Low level applications
- High level applications
- Proven operation procedures

People involved

- PhD. Stefan Simrock - DESY
  - Specification, specification acceptance, operator's experience, testing and evaluation
- PhD. Valeri Ayvazyan - DESY
  - Specification, specification acceptance, operator's experience, testing and evaluation
- PhD. Markus Hoffmann - DESY
  - PR at DESY + early evaluation of chosen design principles
- M. Sc. Bogusław Kosęda - DMCS
  - Specification, partitioning, implementation of the FSM's and planner+EH, formal verification, testing and evaluation.
- M. Sc. Adam Piotrowski - DMCS
  - Implementation of chosen tools needed for automation (e.g integration with DOOCS dashboards)
- M. Sc. Bartłomiej Świercz - DMCS
  - Implementation of chosen tools needed for automation (elements of the FSM framework – if needed)

Time schedule

- 70% Functionality implemented in Dec. 2008
- Evaluation at FLASH – the middle of 2009