DIAGNOSTICS AND AUTOMATION

Magnoni Luca
University of California Irvine (US)

12th March, 2013

Outline

AUTOMATION TODAY

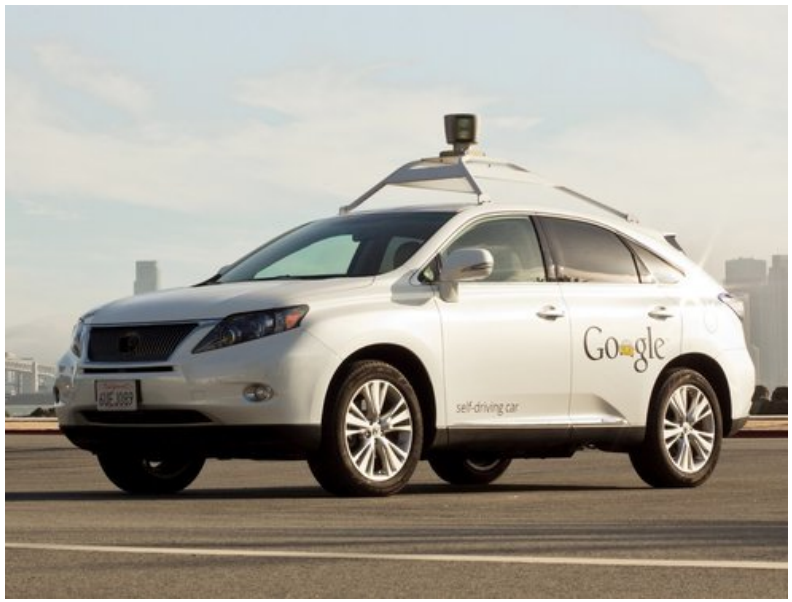
- Not here yet
- but...

APPLE SIRI

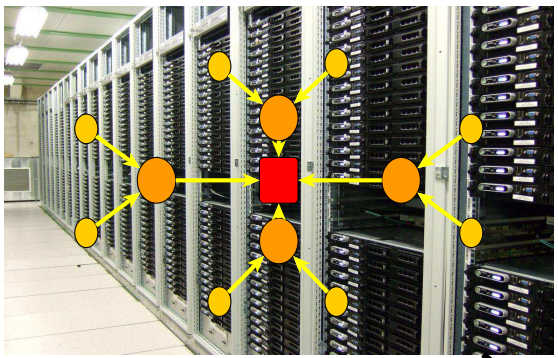
- Automated assistant to do things and to sort out problems
- It has a knowledge base of known situations
- Ask for user inputs to build a representation of the problem
  - *Siri, why my bluetooth headset is not working?*

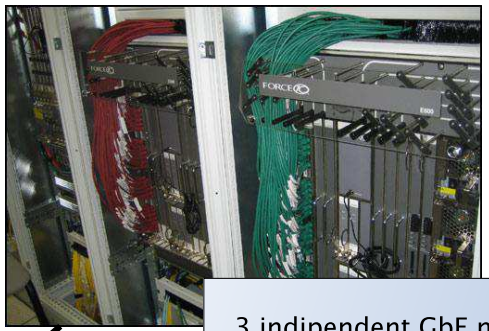
GOOGLE SELF-DRIVING CAR: SO FAR 300.000 MILES WITHOUT ACCIDENT

- Multiple information streams at high rate
- Continuous processing and correlation of information
- React on system evolution over time
  - *wheel increase rpm - car sliding - act on brake*



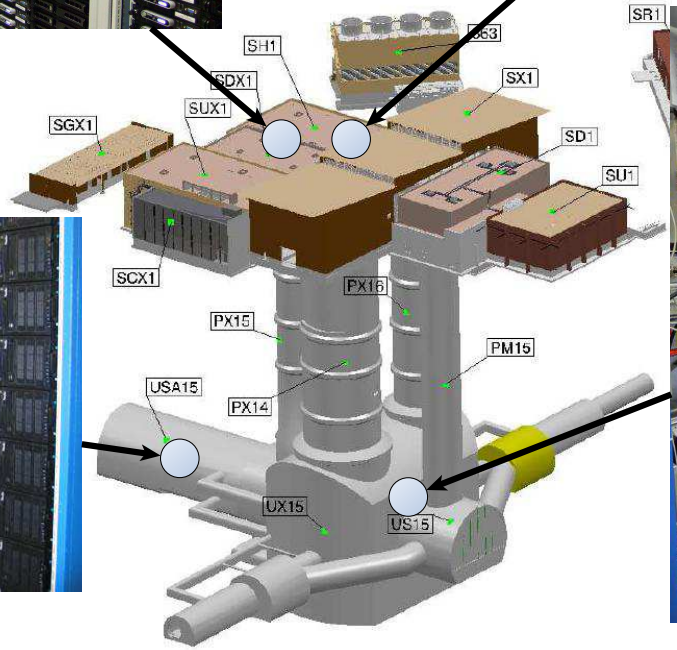
DAQ SYSTEMS: COMPLEX, DATA-INTENSIVE, TIME-CRITICAL INFRASTRUCTURE



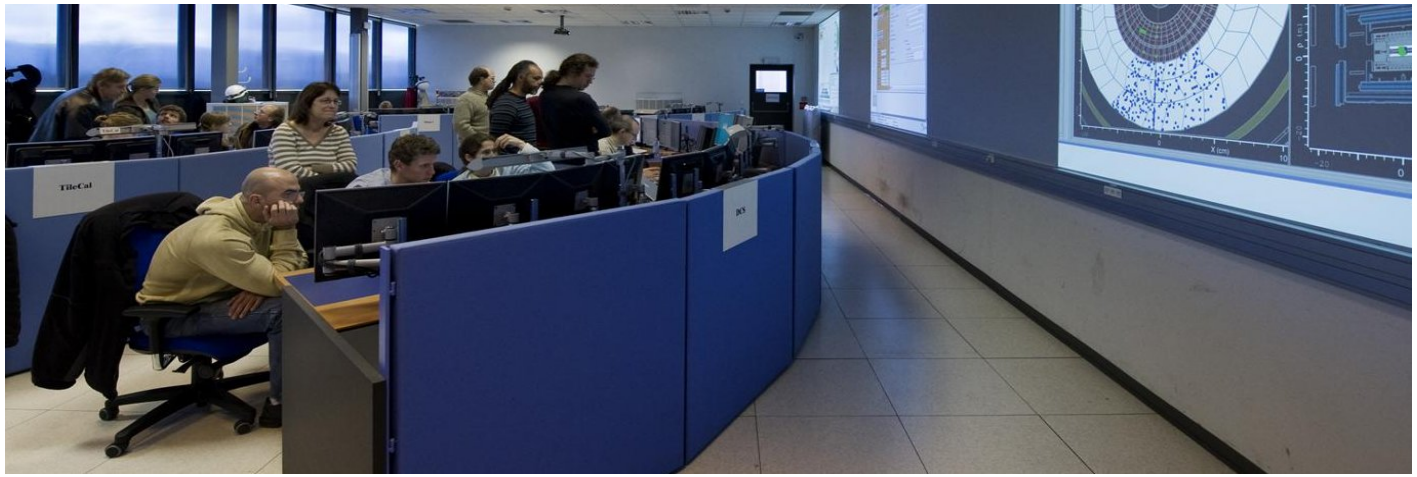


3 indipendent GbE networks
> 100.000 channels

> 12.000 cores
> 20.000 software applications
running on SLC Linux



DAQ SYSTEMS: DRIVEN BY OPERATORS



- Non-expert shift crew, assisted by a set of experts providing knowledge for specific components
- About the **50% of the data taking inefficiency**(ATLAS 2011,p.41-45) is due to **situations where human intervention is involved**
- Help operators with effective tools to understand and troubleshoot problems

AUTOMATION FOR DAQ SYSTEMS?

- Sure, we have it already
- But we want more
- Different tools and approaches are possible
  - "It's time to have the robot doing the work for you..." *Roomba advertisement*

3 application domains

- **Automatize controls and procedures**
- **Error management and recovery**
- **Automated diagnosis, anomaly detection and prompt notification**

AUTOMATION IN DAQ SYSTEMS

- Heterogeneous control systems
  - PVSS/SMI++, same basic infrastructure for detector control systems and DAQ controls (LHCb, Alice)
  - Custom software, WSDL/SOAP over HTTP (CMS)
  - Custom software + expert system/event processing frameworks (ATLAS)
- But not surprisingly, there are many similarities

CONTROLLING THE SYSTEM

- **Finite State Machine (FSM)** management
  - control units automatically handle the status of controlled entities
- **LHC state handling**
  - High Voltages management, LHC clock handshaking
- **Autopilot capability (LHCb)**: control unit autonomously bring the system to data taking status, in normal condition

ERROR MANAGEMENT AND RECOVERY

- **Automated disabling/re-inclusion of faulty part of the system read-out** (Alice, ATLAS, CMS, LHCb)
  - recovery procedure allows for a reconfiguration of the detector's front-end electronics, by pausing the trigger, releasing the readout link, executing a set of predefined action, restarting the read-out link and the trigger
- **Automated detection of host failures**
  - disabling/enabling of computers (ATLAS, LHCb, Alice)
  - or generating new configuration (CMS)
- **Automated restart of failing applications** (ATLAS)

LOGS, ERRORS AND ALARMS

- Alice **infoLogger**: log messages generated by the different subsystems (DAQ, HLT, TRG, detectors, etc) and handled by a common logging framework. The log messages viewer allows to filter by different criteria. Normally the shifter only looks at the higher level messages
- ATLAS: common logging/error framework, ability to subscribe/filter/visualize real time and historical messages. Alerts (log *on steroids*) generated by the Shifter Assistant
- CMS: a **complete log/error/alarms framework**, applications have the ability to report error, fire and retract alarms and operators to ack alarms handled
- LHCb **PVSS II alarming tools**, allowing equipment to generate alarms, archiving , filtering, summarizing and display allowing user to mask and acknowledge

ADVANCED DIAGNOSTICS: INTELLIGENT CENTRAL TOOLS

- End Of Run cause estimation done by the Experiment Control System (Alice)
- **DAQ Doctor** (CMS):
  - Get data from monitoring framework
  - Give advice through custom output
  - **Dump monitoring information in case of problems**
  - **Generate new configuration**
- **Shifter Assistant** (ATLAS):
  - **Correlate error, operational data and farm metrics**
  - Produce alerts to operators
  - Based on **event processing** framework
- Artificial intelligence in the service of system administrators (LHCb)

EVOLUTION

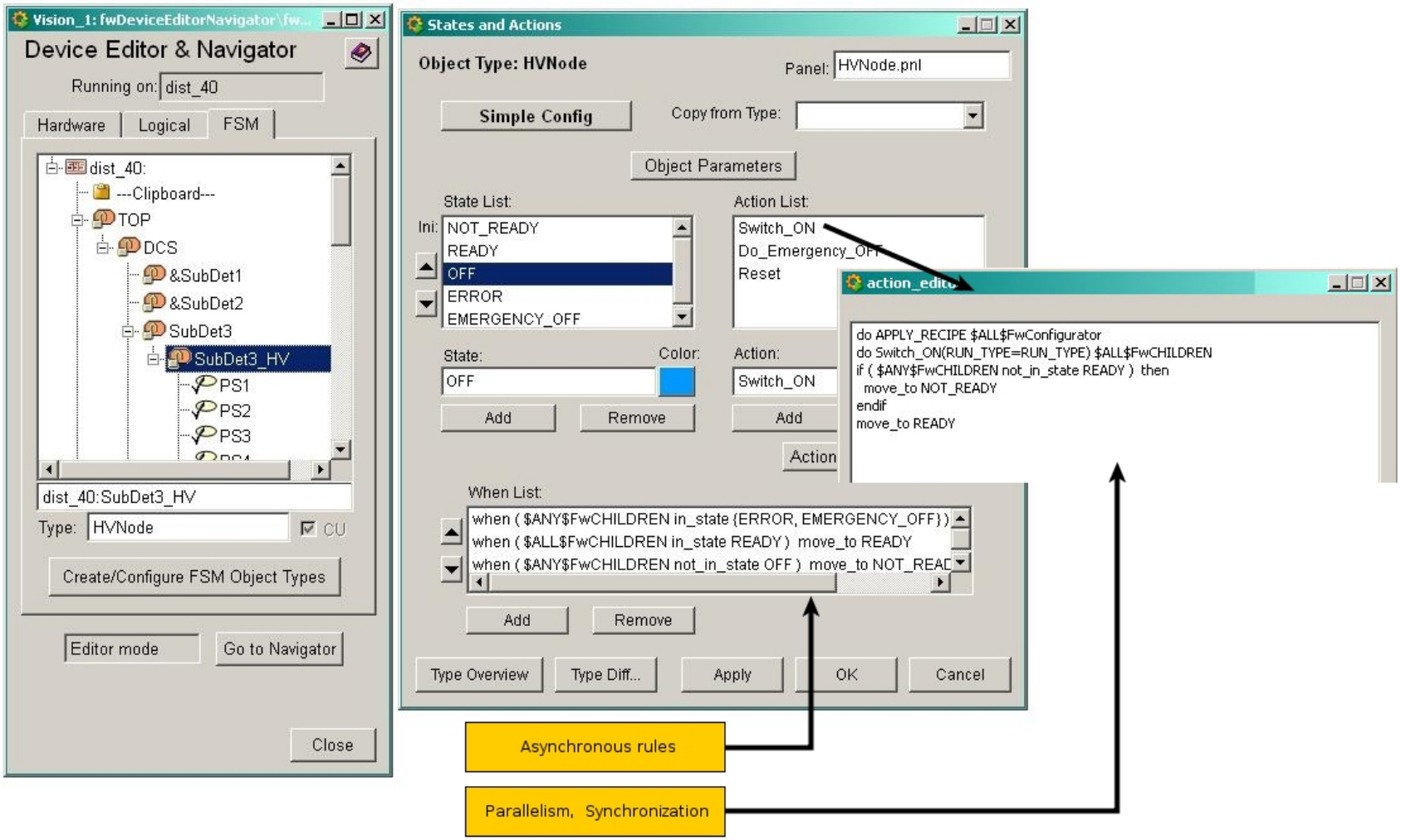
More automation!

- Alice: *expert system to analyze log messages to identify/suggest corrective actions to be applied after End of Run*
- CMS: *more active DAQ Doctor actions, possibly evaulating expert systems framework*
- ATLAS: *evolve error management system to detect more complex situation , event processing*

RULE BASED SYSTEMS

- Actions triggered in case of predefined conditions
- Non-procedural asynchronous execution
- Knowledge base of rules updated with new requirements and experience

SMI++

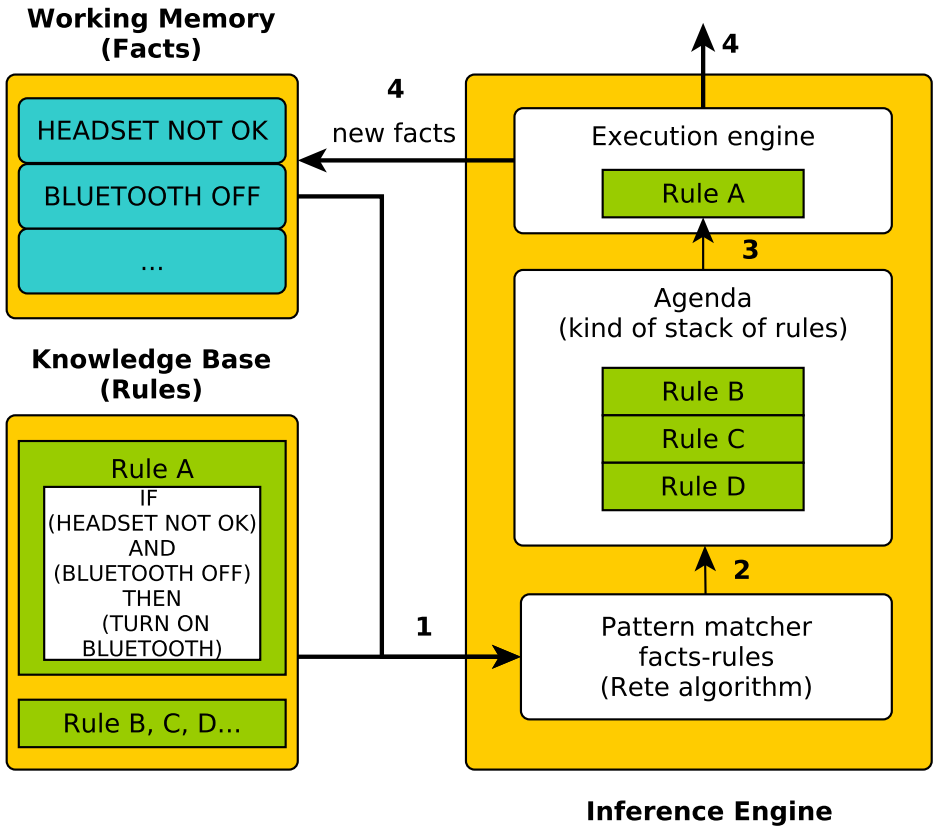

Vision_1: fwDeviceEditorNavigator\fw...
Device Editor & Navigator
Running on: dist_40
Hardware
Logical
FSM
dist_40:
---Clipboard---
TOP
DCS
&SubDet1
&SubDet2
SubDet3
SubDet3_HV
PS1
PS2
PS3
dist_40:SubDet3_HV
Type: HVNode
CU
Create/Configure FSM Object Types
Editor mode
Go to Navigator
Close
States and Actions
Object Type: HVNode
Panel: HVNode.pnl
Simple Config
Copy from Type:
Object Parameters
State List:
Ini:
NOT_READY
READY
OFF
ERROR
EMERGENCY_OFF
Action List:
Switch_ON
Do_Emergency_OFF
Reset
State:
OFF
Color:
Action:
Switch_ON
Add
Remove
Add
Action
When List:
when (ANYFwCHILDREN in_state (ERROR, EMERGENCY_OFF))
when (ALLFwCHILDREN in_state READY) move_to READY
when (ANYFwCHILDREN not_in_state OFF) move_to NOT_READ
Add
Remove
Type Overview
Type Diff...
Apply
OK
Cancel
action_edito
do APPLY_RECIPE ALLFwConfigurator
do Switch_ON(RUN_TYPE=RUN_TYPE) ALLFwCHILDREN
if (ANYFwCHILDREN not_in_state READY) then
move_to NOT_READY
endif
move_to READY
Asynchronous rules
Parallelism, Synchronization

RULE BASED EXPERT SYSTEM (ES)

- Originated from 70s/80s AI research
- Modular knowledge structured in **IF ... THEN... ELSE...** rules
- Data stored as **facts**
- Deductively (forward-chaining) or inductively (backward-chaining)
- ES frameworks
  - CLIPS/JESS Lisp-like syntax, using prefix notation and parentheses to delimit commands and constructs
  - Easy embeddable: C/C++ (CLIPS), Java (JESS, Drools)
  - CLIPS used by ATLAS for the error recovery and diagnostics systems

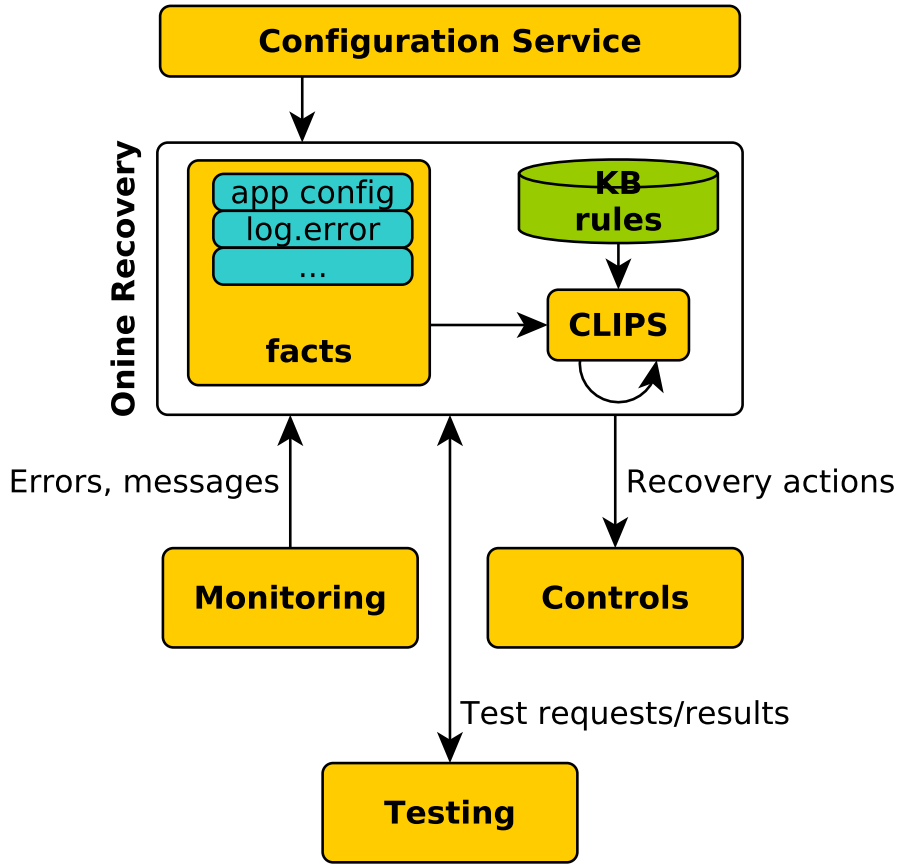
ES ARCHITECTURE: INFERENCE CYCLE

1. Match facts with rules
2. Decide rules order and resolve conflict
3. Execute the top rule RHS, which can infer new fact



ATLAS ONLINE RECOVERY AND DIAGNOSTICS

1. Embedded in local control unit + central instance
2. Represent system condition as facts
3. Interact with DAQ services to automated recovery actions



RECOVERY EXAMPLE

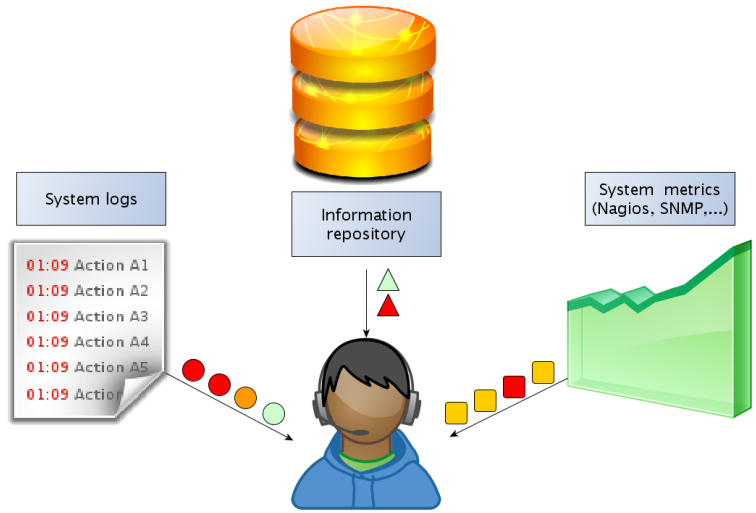
```
;-------------------------------------------------------------------------
; RULE auto-disable-application-died
; Antecedents: RC in RUNNING, application DIED unexpectedly,
; the application belongs to a class that can be autoDisabled
; Consequence: the application is disabled with RC and supervisors,
; the application is restarted and reinserted
;-------------------------------------------------------------------------
(defrule auto-disable-application-died
  ?problem<-(object (is-a APPLICATION-PROBLEM)(TYPE APPLICATION_DIED_UNEXPECTEDLY)
        (OBJECT ?name)(HANDLED FALSE))
  ?app<-(object (is-a OR-APPLICATION)(UID ?name)(CID ?cid)(CLASS-NAME ?cname)
        (RUNS-ON ?host)(MEMBERSHIP IN))
  (object (is-a COMPUTER) (name ?host) (MEMBERSHIP IN)(TEST-STATUS PASSED))
  ?ctrl<-(object (is-a CONTROLLER)(STATE RUNNING)(TRANSITION NONE))
  =>
  (ers-log "[auto-disable-application-died] fired for" ?name)
  (if (send [recover-info] check-restart ?app)
      then
      (ers-debug 0 "[auto-disable-application-died] '" ?name "' on host " ?host "
        is dead.  Recovery action is being taken")
      (recover-app ?app)
      (send ?problem put-HANDLED TRUE)
      (send ?problem put-DECISION DF-RECOVERY)
      else
      (disable-app ?cid)
      (send ?problem put-HANDLED TRUE)
      (send ?problem put-DECISION Ignore)
      )
  )
```

WHEN TO USE EXPERT SYSTEM

- Narrow, well known domain
- Situation matches with fact representation
- No temporal correlation
- No learning ability
- Beware of the dog:
  - depending on the framework, rules may be difficult to test
  - maintenance complexity quickly grows with the number of rules

LIMITATIONS OF THE CONTINUOUS MONITORING

- The meaningful information is not in the single event, but in the overall system behaviour over time
- Operators have to manually correlates and aggregates monitoring data
- Advanced automated tools should be able to replicate this correlation and aggregation



Diagnostics via stream processing

- Many information sources (log streams, information system, farm metrics, ...)
- Several technologies to gather data (DB, API library, ...)
- Heterogeneous data (format, publication mechanism)
- Dynamic system conditions (calibration, physics, ...)

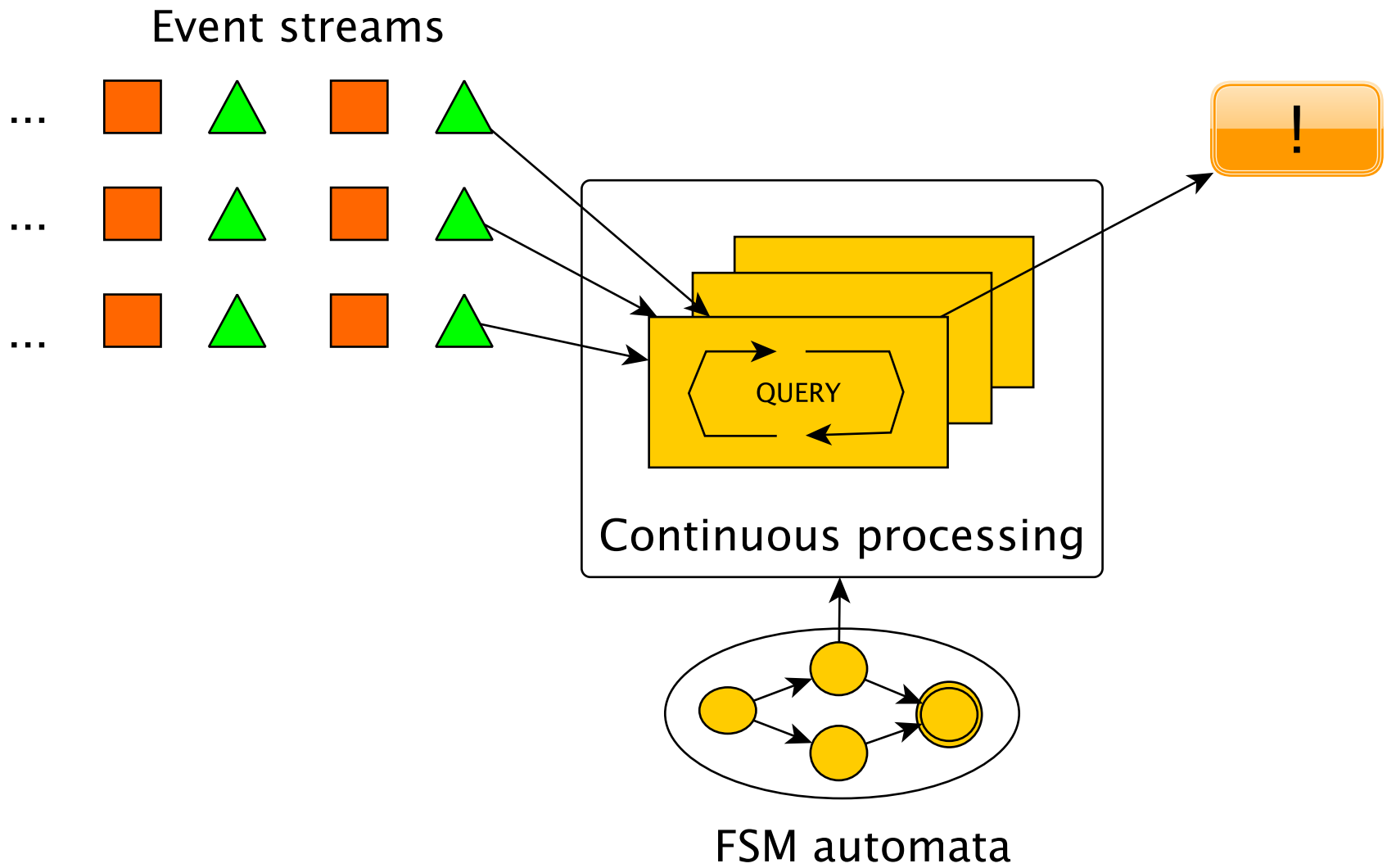
Main challenges

1. **Information gathering** (different streams with thousands of information update per second)
2. **Information processing** (building Knowledge Base, process data streams and discover complex patterns)
3. **Present results** in several way

COMPLEX EVENT PROCESSING (CEP)

- A set of technologies to **process events and discover complex patterns among streams of events**
- A cross between Data Base Management Systems and Rule Engines
- The distinguishing characteristics are:
  - support for time based boundaries, fixed size moving windows, aggregation and partitioning/grouping events on a streams of event
  - expressed via **pattern languages derived from the SQL**
  - augmented **with constructs to express event relationships**: time, cause and aggregation
  - with **streams replacing tables** in a continuous evaluation model

CEP PROCESSING MODEL

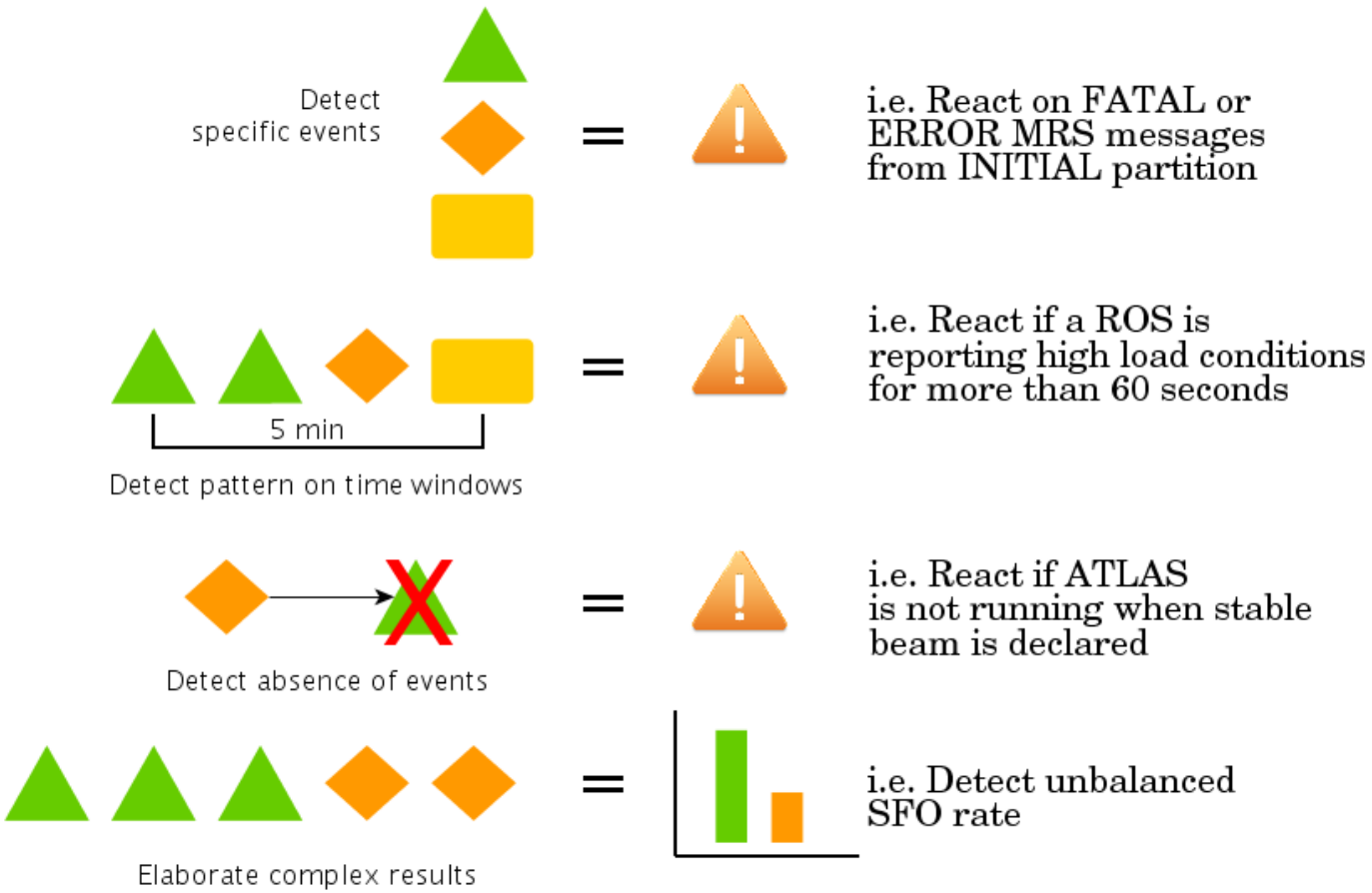


CEP FRAMEWORKS

- Esper: most widely adopted open-source CEP solution
  - used in ATLAS for the Shifter Assistant project
  - and in CMS for the CSC *expert system* prototype
- Oracle offers its own product: Oracle CEP
- Many other non open-source frameworks (i.e. StreamBase, SoftwareAG)

Growing interest by different CERN communities: *Data Analytics* workshop, *New data management* seminar

PATTERNS


Detect
specific events
=
i.e. React on FATAL or
ERROR MRS messages
from INITIAL partition
5 min
=
i.e. React if a ROS is
reporting high load conditions
for more than 60 seconds
Detect pattern on time windows
=
i.e. React if ATLAS
is not running when stable
beam is declared
Detect absence of events
=
i.e. Detect unbalanced
SFO rate
Elaborate complex results

EPL PATTERNS

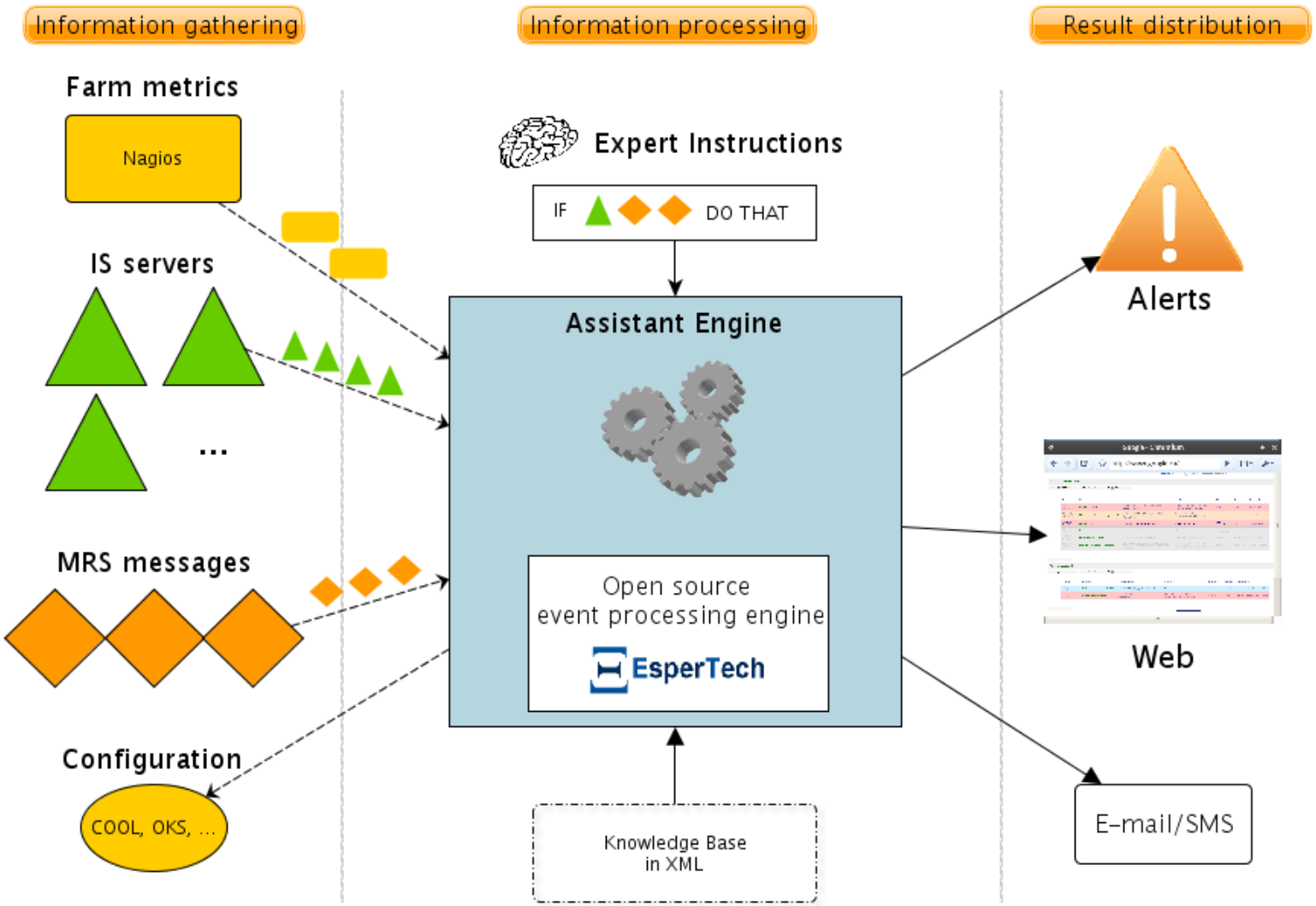
- Single event
  - `every` **`Message`**`(messageID=xyz)`
- Data window and filtering
  - `select * from` **`Message(severity='Error').win:time_batch(30 sec)`**
- Aggregation
  - `select messageID, messageTxt,` **`sum`**`(numberOfMessages) from Message.win:time_batch(30 sec)` **`group by`** `messageID, messageTxt`
- Database integration, message queuing systems, custom software extensions

CEP application: ATLAS Shifter Assistant

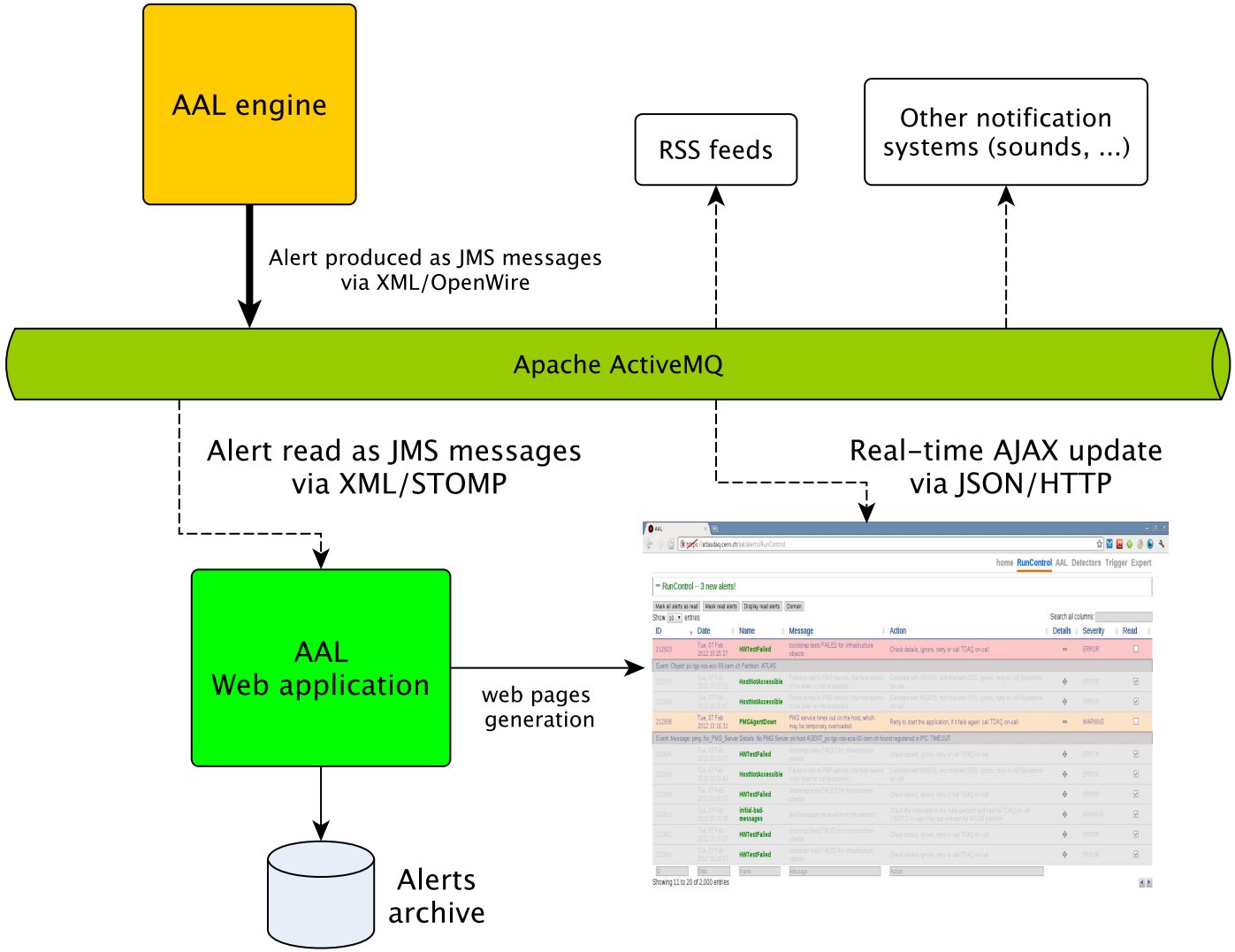
The Assistant

- A tool meant at guiding the operator in his daily work. It can both help diagnosing problematic situations and suggesting actions to take, as well as remind the operator that he should (not) do something.
- DAQ systems already provide all the information, it is a matter of using it effectively.

SHIFTER ASSISTANT ARCHITECTURE



SHIFTER ASSISTANT ARCHITECTURE CONTD.



CONCLUSION AND LESSON LEARNED

- We like automation and we want more
  - but it means more than automatic reactions
  - the ability to detect **complex data patterns over time** allows for a **new generation of automated tools**
- Expert-system solutions appropriate for narrow domain and well-known conditions
- Event-processing more suited for continuous data processing and complex pattern detection
  - with **many potential applications** in complex distributed systems such as DAQs
- Do not reinvent the wheel. Time spent studying new technologies pays back
  - more will come, with the focus on distributed systems and big data as today (real-time Hadoop?)

CONCLUSION AND LESSON LEARNED

- When developing an automated system targeting human operators is fundamental to have its functionalities accepted and understood from the early phase
  - **for operators to trust it**
  - **for experts to be aware of the new capabilities**
- Keep operators and experts in the development loop
- Revise requirements often
  - waterfall approach is simply not the best for this sort of new unknown tools
- One direction for the future:
  - **from automated to intelligent**: make the system learn

THANK YOU! ANY QUESTION?