OpsPilot

Alarm Rationalisation — User Manual

Reduce Alarm Flooding · EEMUA 191 · AI Engineering Co-Pilot

What this guide covers — what alarm rationalisation is, how the OpsPilot module applies EEMUA 191, what to have ready, and the register you receive.

1. What is alarm rationalisation?

Alarm rationalisation cures alarm flooding — the condition where operators get so many alarms that they stop trusting any of them, acknowledging and ignoring rather than responding. That erosion of vigilance is dangerous: the one alarm that matters is lost in the noise. Rationalisation works through the alarm system and decides, for each alarm, whether it earns its place.

OpsPilot uses the *EEMUA 191 methodology* — the recognised standard — including the five-question rationalisation test applied to each alarm.

2. What the OpsPilot module does

Role	Responsibility
🎯 **AI Coach (OpsPilot)**	Guides a structured rationalisation per EEMUA 191 — identifying bad actors, applying the five-question test to each alarm, recommending the right action (keep, modify, delete or automate), and building an MOC-ready rationalisation register.
👷 **Control / Operations Engineer (you)**	Provide the alarm data and the operational context — you know which alarms operators actually respond to and which are nuisance alarms eroding vigilance.

3. How it works — the process

#	Stage
1	Alarm system context — DCS/SCADA platform, current alarm rate
2	Alarm data collection — frequency, acknowledgement time, priority
3	Bad actor identification — top 10 by activation count
4	EEMUA 191 rationalisation test — five questions per alarm
5	Setpoint and priority review
6	Rationalisation register — Keep / Modify / Delete / Automate

#	Stage
7	Word report

4. What you will be asked — have this ready

- The DCS, SCADA or PLC platform that manages alarms.
- The current alarm rate — alarms per operator per hour.
- Alarm data — activation frequency, acknowledgement time and current priority.
- The operational context — which alarms operators actually act on.

5. What you receive — the output

An Alarm Rationalisation Register and report (Word): the system context and alarm rate, the bad-actor list, the EEMUA 191 test applied to each, a setpoint and priority review, and an MOC-ready register recommending Keep / Modify / Delete / Automate for every alarm.

6. Worked example (illustrative)

An operator is receiving 200 alarms an hour — far above the EEMUA target, so they've stopped reading them. The bad-actor analysis shows the top ten alarms generate most of the flood: one is a level alarm chattering around its setpoint, another duplicates a trip that already alarms. The five-question test on the chattering alarm finds it requires no operator action, so it's modified (deadband added) or deleted; the duplicate is deleted. Clearing just the top ten bad actors can drop the rate dramatically — and every change is captured in an MOC-ready register, because changing an alarm system is itself a management-of-change activity.

7. Getting the best result

- **Start with bad actors.** A handful of alarms usually cause most of the flood.
- **Apply the test honestly.** If an alarm needs no operator action, it isn't an alarm.
- **Keep / Modify / Delete / Automate.** Every alarm gets a deliberate decision, not the benefit of the doubt.
- **Route changes through MOC.** Changing alarms is a controlled change — the register is built for it.