CERTIFIED
MANAGEMENT SYSTEMS
CERTIQUALITY
UNI EN ISO **9001:2015**
UNI CEI EN ISO/IEC **27001:2022**
UNI EN ISO **45001:2023**
UNI CEI EN ISO **50001:2018**

Elettra Sincrotrone Trieste

Elettra
Sincrotrone
Trieste

CERTIFIED
MANAGEMENT SYSTEMS
CERTIQUALITY
UNI EN ISO 9001:2015
UNI CEI EN ISO/IEC 27001:2022
UNI EN ISO 45001:2023
UNI CEI EN ISO 50001:2018

Modernizing Starter
Parallel startup, configurable arguments, and what's next

Q&A at slido.com #7591860

Alessio Igor Bogani · Elettra Sincrotrone Trieste

Elettra Sincrotrone Trieste

Starter at a glance

- ✓ The Tango device server that supervises every other device server on a host
- ✓ The backend behind Astor — the GUI everyone uses
- ✓ On the critical path of every boot of every Tango-controlled host
- ✓ Today: three modernization changes and one experimental side project

Elettra Sincrotrone Trieste

Agenda

- ✓ 1. Parallel startup — branch parallel-servers-startup (ready for review)
- ✓ 2. Per-server arguments — community patch from lrlunin (TU Berlin)
- ✓ 3. Replacing the /proc walk in CheckProcessUtil — design phase
- ✓ 4. Teaser: Starter as PID 1 — branch starter-as-init

Elettra Sincrotrone Trieste

§1 — Parallel startup: the problem

- ✓ DevStartAll walks startup levels in order, one server at a time
- ✓ Within each level: fork one server, sleep, fork the next
- ✓ The sleep was TIME_BETWEEN_STARTUPS — a hard-coded 500 ms macro
- ✓ 30 servers at the same level → 15 s of pure sleep, every boot
- ✓ Time scales with the sum of per-server start times, not with the slowest

Elettra Sincrotrone Trieste

§1 — The change (branch parallel-servers-startup)

- ✓ TIME_BETWEEN_STARTUPS macro → InterStartupServerWait device property
  - Default 500 ms; set to 0 to disable inter-server throttling
- ✓ New boolean: EnableParallelStartup (default false)
  - When true: fork every server at a level back-to-back, then poll them concurrently
- ✓ Refactor: poll_status() helper extracted from StartProcessThread::run
  - Same behaviour as before — just the building block for the parallel path
- ✓ Default off → Astor users see no surprises; opt-in only

Elettra Sincrotrone Trieste

§1 — Impact

- ✓ Sequential (current default)
  - Per-level wall-clock = Σ start_i
  - Plus (N−1) × InterStartupServerWait
  - Worst offender: a slow server blocks every server behind it
- ✓ Parallel (opt-in)
  - Per-level wall-clock ≈ max(start_i)
  - Scales with the slowest server, not their sum
  - Same end state, dramatically lower boot time on busy hosts

Elettra Sincrotrone Trieste

§1 — Parallel shutdown: make it symmetric

- Startup is now parallel — but DevStopAll is still strictly sequential
- DevStop issues a blocking Kill, one server at a time → shutdown costs the sum of every round-trip
- On a big host, stopping can now be slower than starting
- Proposal: reuse the startup pattern — act on all, then watch all
- Fire Kill to every server without blocking (thread pool / async Tango), then poll until down
- Reuse the switches (EnableParallelStartup, InterStartupServerWait); not built yet — decide now

Elettra Sincrotrone Trieste

§2 — Per-server arguments (Irlunin, TU Berlin)

- ✓ Today, Starter builds argv = [server_name, instance_name] — and nothing else
- ✓ Want a verbosity flag, a custom log target, any extra option? Wrap the binary in a shell script
- ✓ On Irlunin's GitLab fork, branch args-forward (commit 385f668)
- ✓ Adds one Tango device property per controlled server
  - Property name = <server>_<instance> (slash replaced by underscore)
  - Value = string list, appended to argv at fork time

Elettra Sincrotrone Trieste

§2 — How it works & what's missing

- ✓ init_device: Starter reads the properties and attaches them to its ControlledServer entries
- ✓ Fork site (StartProcessThread): values are appended to argv before execvp
- ✓ Per-server tuning lives in the Tango database — alongside the rest of the configuration
- ✓ Missing before merge:
  - Windows path not yet updated
  - No environment-variable support
  - A couple of small memory-management cleanups
- ✓ Otherwise: the design is exactly what we want — I'd like to discuss merging it

Elettra Sincrotrone Trieste

§3 — Replacing the /proc walk in CheckProcessUtil

- ✓ Today on Linux: open /proc, walk every numeric directory, read /proc/<pid>/cmdline as text — every poll
- ✓ Cost: O(total processes on the host), plus text parsing on every tick
- ✓ Races against PID reuse; detection latency ≥ 1 s for a crashed server
- ✓ Goal: event-driven crash detection, kernel-native
- ✓ Candidates under evaluation:
  - pidfd_open — stable per-process handles
  - netlink process events connector — push-style fork/exec/exit notifications
  - eBPF tracepoints on sched_process_* — most flexible, highest barrier
- ✓ Trade-offs: kernel version, required capabilities, portability

§4 — Teaser: Starter as PID 1 (starter-as-init)

- ✓ Personal experimental branch — Linux-only
- ✓ Optional CMake flag STARTER_AS_INIT — off by default, opt-in at build time
- ✓ When set: Starter can run as PID 1 in a minimal VM or container
  - Mounts /proc, /sys, /dev/pts, /run, /tmp, /dev/shm
  - Async-signal-safe SIGCHLD reaper for zombie children
  - SIGTERM-driven graceful shutdown via the Tango event loop
  - OOM-protected (oom_score_adj = -1000), stdio redirected to /dev/console
- ✓ Far from production — TODO.md is long (PID-tracking, respawn policy, cgroups, watchdog…)

Elettra Sincrotrone Trieste

Summary

- ✓ Two patches ready for review:
  - parallel-servers-startup (mine) — opt-in concurrent startup, plus symmetric shutdown
  - args-forward (Irlunin / TU Berlin) — per-server arguments via Tango properties
- ✓ One project under design: event-driven crash detection
- ✓ One wild idea: Starter as PID 1
- ✓ Discussion welcome — let's pick this up at the meeting and on GitLab

CERTIFIED
MANAGEMENT SYSTEMS
CQY
CERTIQUALITY
UNI EN ISO 9001:2015
UNI CEI EN ISO/IEC 27001:2022
UNI EN ISO 45001:2023
UNI CEI EN ISO 50001:2018

Thank you!