CHANGELOG — v0.6.2 Modular Code Cleanup

Date: February 25, 2026 **Author:** Justin Kwarteng **Type:** Refactoring (zero behavior changes) **Status:** ☑ Complete — all builds passing, verified on hardware

Summary

Split all monolithic source files across the BLE Mesh project into focused, single-responsibility modules. This refactoring touched **5 codebases** (4 ESP firmware projects + 1 Python gateway), converting **~4,500 lines** of monolithic code into **26 focused module files** while preserving identical runtime behavior.

ESP Firmware Changes

Batch 1 — Sensor Node (`ESP/ESP-Mesh-Node-sensor-test/`)

Before: `main.c` (781 lines) — everything in one file

After: `main.c` (59 lines) — thin orchestrator calling module init functions

New File	Responsibility	Key Functions
`sensor.c` / `sensor.h`	I2C bus init, INA260 probe, voltage/current reads	`sensor_init()`, `sensor_is_ready()`, `ina260_read_voltage()`, `ina260_read_current()`
`load_control.c` / `load_control.h`	PWM output, duty cycle, ramp test logic	`pwm_init()`, `set_duty()`, `start_ramp()`, `stop_load()`, `ramp_task()`
`command.c` / `command.h`	Serial console parser, command dispatch	`console_task()`, `process_command()`
`mesh_node.c` / `mesh_node.h`	BLE Mesh provisioning callbacks, vendor model, Generic OnOff server	`ble_mesh_init()`, `provisioning_cb()`, `generic_server_cb()`, `custom_model_cb()`
`nvs_store.c` / `nvs_store.h`	NVS handle + device UUID globals	`NVS_HANDLE`, `dev_uuid`

Deleted:

- `ble_service.c` — legacy GATT service code from v0.3.0 (unused since mesh migration)
- `ble_service.h` — corresponding header (dead code, not in CMakeLists.txt)

CMakeLists.txt: Updated SRCS list to include all new `.c` files.

Batch 2 — GATT Gateway (`ESP/ESP_GATT_BLE_Gateway/`)

Before: `main.c` (1,036 lines)

After: `main.c` (89 lines) — preserves critical GATT-before-mesh init order

New File	Responsibility	Key Functions
`gatt_service.c` / `gatt_service.h`	Custom GATT service registration, advertising, notification send	`gatt_register_services()`, `gatt_start_advertising()`, `gatt_notify_sensor_data()`
`mesh_gateway.c` / `mesh_gateway.h`	Mesh composition, callbacks, vendor model client, command forwarding	`mesh_init()`, `send_vendor_command()`, `send_mesh_onoff()`, `custom_model_cb()`
`command_parser.c` / `command_parser.h`	Parse "NODE:CMD:VAL" format from Pi 5 GATT writes	`parse_command()`, `execute_command()`
`monitor.c` / `monitor.h`	Continuous sensor polling task	`monitor_task()`, `start_monitoring()`, `stop_monitoring()`
`node_tracker.c` / `node_tracker.h`	Track provisioned node addresses for routing	`node_tracker_add()`, `node_tracker_get_addr()`
`nvs_store.c` / `nvs_store.h`	NVS handle + device UUID globals	`NVS_HANDLE`, `dev_uuid`

CMakeLists.txt: Updated SRCS list to include all new `.c` files.

Batch 3 — Provisioner (`ESP/ESP-Provisioner/`)

Before: `main.c` (868 lines)

After: `main.c` (51 lines)

New File	Responsibility	Key Functions
`mesh_config.c` / `mesh_config.h`	Mesh composition, model arrays, provision struct	Mesh model/element definitions, config constants
`node_registry.c` / `node_registry.h`	Track provisioned devices by UUID/address	`node_registry_add()`, `node_registry_get()`
`composition.c` / `composition.h`	Parse device composition data after provisioning	`store_comp_data()`, `get_comp_data()`
`model_binding.c` / `model_binding.h`	AppKey add + model bind sequence	`bind_app_key()`, `bind_model()`
`provisioning.c` / `provisioning.h`	Provisioning callbacks, config client, mesh init	`ble_mesh_init()`, `provisioning_cb()`, `config_client_cb()`, `recv_unprov_adv_pkt()`

CMakeLists.txt: Updated SRCS list to include all new `.c` files.

Batch 4 — Relay Node (`ESP/ESP-Mesh-Relay-Node/`)

Before: `main.c` (311 lines)

After: `main.c` (57 lines)

New File	Responsibility	Key Functions
`mesh_relay.c` / `mesh_relay.h`	Mesh composition (relay-enabled), provisioning callbacks	`ble_mesh_init()`, `provisioning_cb()`, `config_server_cb()`
`led.c` / `led.h`	LED heartbeat task (fast=unprovisioned, slow=active)	`led_init()`, `led_heartbeat_task()`
`nvs_store.c` / `nvs_store.h`	NVS handle + device UUID globals	`NVS_HANDLE`, `dev_uuid`

CMakeLists.txt: Updated SRCS list to include all new `.c` files.

Python Gateway Changes

Gateway TUI (`gateway-pi5/`)

Before: `test-13-tui.py` (1,722 lines) — all classes in one file

After: 7 focused modules, `test-13-tui.py` kept as rollback reference

New File	Lines	Responsibility
`constants.py`	17	UUIDs, regex patterns, device name prefixes
`ble_thread.py`	68	Dedicated asyncio event loop thread for bleak BLE I/O
`node_state.py`	19	`NodeState` dataclass (duty, voltage, current, power, responsiveness)
`power_manager.py`	540	`PowerManager` — equilibrium-based power balancer with priority weighting
`dc_gateway.py`	480	`DCMonitorGateway` — BLE scanning, GATT connection, notification parsing, CLI mode
`tui_app.py`	470	`MeshGatewayApp` — Textual TUI with sidebar, DataTable, RichLog, command dispatch
`gateway.py`	140	Entry point — argparse, TUI/CLI mode selection

Entry point changed: `python gateway.py` (was `python test-13-tui.py`)

Circular import handling: `power_manager.py` uses `TYPE_CHECKING` guard for `DCMonitorGateway` type hint.

`_HAS_TEXTUAL` pattern: Duplicated in `dc_gateway.py` (for log routing) and `gateway.py` (for TUI/CLI fallback). `tui_app.py` imports textual unconditionally since it's only loaded when textual is available.

Build & Verification

ESP Firmware

All four projects verified with:

```
idf.py fullclean
idf.py set-target esp32c6
idf.py build
```

Python Gateway

All seven modules verified with:

```
python -c "import py_compile; py_compile.compile('<file>', doraise=True)"
```

All pass syntax compilation. Import chain verified (fails only on `bleak` which is Pi 5-only).

Hardware Verification

Deployed to hardware and confirmed:

- Mesh provisioning works identically
- Sensor data flows through vendor model
- GATT gateway bridges Pi 5 ↔ mesh
- Power management balancing works
- No reprovisioning required

What Did NOT Change

- **Zero behavior changes** — every function does exactly what it did before
- **No protocol changes** — same BLE Mesh opcodes, same GATT UUIDs, same NVS keys
- **No reprovisioning needed** — devices boot with existing mesh state
- **No dependency changes** — same ESP-IDF version, same Python packages

---

File Statistics

Metric	Before	After
ESP monolithic files	4 × `main.c` (3,006 lines total)	4 × thin `main.c` (~256 lines) + 19 modules
Python monolithic files	1 × `test-13-tui.py` (1,722 lines)	7 focused modules (1,734 lines)
Total module files created	0	26 (19 C pairs + 7 Python)

Metric	Before	After
Largest single file	1,722 lines (Python) / 1,036 lines (C)	540 lines (Python) / 447 lines (C)
Legacy files deleted	0	2 (`ble_service.c`/`.h` in sensor node)

Rollback Plan

ESP Firmware

Git history contains the original monolithic `main.c` files. Revert with:

```
git checkout HEAD~1 -- ESP/*/main/main.c
```

Python Gateway

Original `test-13-tui.py` is preserved in-place. Revert with:

```
cp test-13-tui.py gateway.py
```