Cnuas Facility Twin, Datasheet

Early facility prototype, OpenUSD scenes, simplified loads and emulated ORV3 telemetry

Document DS-CNU-030 • **Revision** C • **Issued** 23 August 2026 • **Status** Published document, experimental research preview

Experimental research preview

Facility Twin is in its infancy. Existing code provides an early exploratory prototype for scene generation, simplified load calculations and connections to emulated rack telemetry, not a ready or validated data-centre digital twin. Published is the status of this document, not a product-readiness or source-release certification. Do not use its outputs for engineering, procurement, safety or operational decisions.

Item	Value
Part	`cnuas-facility`
Type	Experimental facility scene and simplified load prototype
Package	`cnuas-facility`, version 0.1.0
Version	`5c3d075-dirty`
Repo	PacketFive/cnuas

1. Overview

Cnuas is an open-source product/platform and an extensible research framework. The overall effort is experimental research, with some implemented prototypes and other capabilities in progress or planned. Its core and accelerator stack are intended for collaborative community development.

The existing Facility Twin prototype generates an illustrative AI and HPC data-centre campus as an OpenUSD scene, calculates simplified load roll-ups and writes readings from an emulated Open Rack V3 power shelf onto scene attributes. These are exploratory software interfaces, not evidence of accurate physical behaviour or a validated campus design.

We intend to make this early prototype available in the public release. This does not promise a finished Facility Twin extension in the initial release or mean that the full source has already been publicly released.

OpenUSD is the scene interchange format. NVIDIA Omniverse and Isaac Sim are intended exploration environments, not a certified compatibility matrix. Successful loading, materials, attributes and telemetry behaviour depend on the viewer and version; compatibility with every OpenUSD viewer is not claimed.

Implemented exploratory features

- Declarative illustrative campus specification, from site dimensions to racks.
- Strict user JSON configuration with template generation and spatial checks before OpenUSD authoring, not validation of an engineering design.
- Three simplified building footprints, the I, H and E shapes.
- Geometric data halls with rack rows, aisle spacing and support rooms.
- Illustrative yards, roads, plant compounds, groves and lakes.
- Assumed-load roll-up from rack to campus, with illustrative plant counts.
- Sampling of the emulated Open Rack V3 shelf over Modbus, written onto the scene as typed attributes; physical-system fidelity remains unvalidated.

2. Module map

Module	Responsibility
`spec.py`	Declarative campus, building and hall specification, and the illustrative reference campus
`config.py`	JSON template, strict loader, custom rack catalogue and spatial checks
`catalog.py`	Assumed equipment dimensions and load ratings per rack and plant type
`plan.py`	Resolves a specification into placed rooms, racks, plant and roads
`layout.py`	Rectangle mathematics, building footprints, rack row layout
`geom.py`	Geometry and material primitives
`usd.py`	Writes the campus OpenUSD scene graph
`schema.py`	The `cnuas:` attribute namespace carried on the scene
`power.py`	Simplified load roll-up and illustrative plant counts
`orv3.py`	Modbus client and register decoding for the power shelf
`live.py`	Samples the shelf and animates the scene
`units.py`	Named dimensional constants
`cli.py`	Command line front end

3. Command reference

Command	Purpose
`build`	Write the campus OpenUSD stage
`power`	Print the power roll-up
`rooms`	Print the room programme
`live`	Sample the Open Rack V3 shelf and animate the stage
`template`	Write the complete reference campus as editable JSON
`validate`	Check custom JSON structure and spatial constraints without writing OpenUSD; not physical scenario validation

4. Scene specification

These reference values are illustrative inputs or targets, not a measured facility, calibrated design or demonstrated operating scenario.

Parameter	Reference value
Campus name	Cnuas Ridge AI Campus
Site footprint	1000 m by 800 m
Buildings	3, one each of the I, H and E footprints
Data halls	16 across the three buildings
Assumed power usage effectiveness	1.15
Illustrative target campus load	Close to 300 MW
Stage root	`/World`
Scene branches	`/World/Looks`, `/World/Site`, `/World/Yards`, `/World/Buildings`, `/World/Prototypes`
Default output format	`.usda`, with `.usdc` and `.usd` also supported

5. Equipment catalogue

Entry	Assumed load or sizing basis
Cnuas Open Rack V3, 24 OpenU	6 kW
GB200 NVL72 class rack	132 kW
Storage rack	12 kW
Network rack	15 kW
Medium voltage switchgear, transformer, generator, coolant distribution unit, fan wall, indirect evaporative cooler, dry cooler	Simplified catalogue and power-model assumptions

6. Power model

The arithmetic below is implemented. It uses simplified illustrative assumptions, not a calibrated electrical or thermal simulation. Catalogue ratings, PUE, heat rejection and plant unit sizes must not be interpreted as validated equipment specifications or used for engineering or operational decisions.

Quantity	Definition
Information technology load	Sum of rack ratings in scope
Facility load	Information technology load multiplied by the power usage effectiveness
Overhead	Facility load less information technology load
Heat rejected	Assumed equal to the information technology load
Density	Load per unit floor area

Roll-up functions run at hall, building and campus scope. Illustrative plant counts are derived from the resulting assumed load; they do not establish capacity adequacy, redundancy, protection or cooling performance.

Plant	Assumed unit size used for illustrative counts
Generator	3000 kW
Cooler	3500 kW
Coolant distribution unit	2000 kW

7. Live telemetry

Item	Value
Source	Emulated Open Rack V3 power shelf
Protocol	Modbus remote terminal unit framing
Transport	Serial line or a TCP endpoint, by default port 3485 on the loopback address
Decoded from the power supply	Voltage, current, power, fan speed, temperature
Decoded from the battery backup	State of charge and unit state
Write back	Typed attributes in the `cnuas:` namespace on the corresponding scene primitives

The shelf itself is emulated by the BMC stack, which runs the power shelf firmware and presents the RS-485 segment.

These connections exercise software data paths from emulated telemetry. They do not establish fidelity to physical shelves, end-to-end fault coverage or validated coupling between rack behaviour and campus electrical or thermal behaviour.

8. Software checks and outstanding scenario validation

The following inventory covers prototype software tests under `facility/`. Passing these checks does not validate facility scenarios, physics, design suitability or product readiness.

Area	Test	Cases
Power roll-up and plant sizing	`tests/test_power.py`	8
Footprints, rectangles and rack rows	`tests/test_layout.py`	19
Scene graph generation, including custom-configuration read-back	`tests/test_usd.py`	14
Shelf sampling and scene animation	`tests/test_live.py`	19
Custom JSON round trip, validation, automatic plant sizing and CLI	`tests/test_config.py`	6

The collected inventory totals 66 software cases. See the Validation Matrix for software-check context, not evidence of a validated facility twin.

Substantial scenario validation remains needed. The following work is not claimed complete.

Area requiring validation	Work still needed
Geometry and catalogue assumptions	Check dimensions, placement constraints, aisle and plant clearances, equipment representations and applicable reference data
Electrical, load and thermal behaviour	Compare assumed rack loads, PUE, heat rejection and plant counts with reference calculations and measurements across operating conditions
Telemetry integration and faults	Assess mapping, units, timing, stale or missing readings, malformed data, disconnects and fault propagation across the integrated scenario
References and calibration	Establish traceable datasets, calibration procedures, uncertainty bounds and acceptance criteria
Held-out scenarios	Evaluate independent scenarios not used to select assumptions or calibrate models, including abnormal and boundary conditions

9. Operating notes

- The prototype represents geometry, simplified load and emulated telemetry. It does not perform computational fluid dynamics, electrical fault studies or structural analysis, and the plant counts are illustrative arithmetic rather than engineering designs.
- Building footprints are limited to the I, H and E shapes in this revision.
- Custom configurations can add rack types, but plant item types remain the built-in electrical and mechanical catalogue in this revision.

10. Integration information

Item	Value
Repo	`PacketFive/cnuas`
Source	`facility/src/cnuas_facility/`
Entry point	`cnuas-facility`
Python	3.10 or later
Dependencies	OpenUSD Python bindings
Viewers	OpenUSD-capable tools; Omniverse and Isaac Sim are intended exploration environments, with viewer/version compatibility requiring validation
Release intent	Make the existing early prototype available publicly, not deliver a finished Facility Twin extension in the initial release

11. Revision history

Revision	Notes
A	First publication. Reference campus, catalogue ratings and plant unit sizes read from the source.
B	Added the beginner workflow and strict custom-campus JSON template, validation, custom rack types, configurable roads and automatic per-building plant counts.
C	Marked experimental research preview, public-release intent and limits of software checks, load assumptions, viewer compatibility and scenario validation.

PacketFive, Packet Five Networks Ltd., Dublin, Ireland.

Cnuas Virtual AI/HPC Infrastructure is published at `https://github.com/PacketFive/cnuas` under the Apache License 2.0; read it at `https://github.com/PacketFive/cnuas/blob/main/LICENSE`.

Cnuas Virtual AI/HPC Infrastructure is emulation software. It is not affiliated with, endorsed by, or derived from any hardware vendor, and every device it models is a software artefact. The work is published by its authors in a personal capacity and is not sponsored or endorsed by any employer.

Specifications describe the referenced revision of the software and may change without notice. Contact info@packetfive.com.