EXECUTIVE CURRICULUM VITAE | 2026

Llewellyn van der Merwe

Principal engineering leadership across software architecture, platform engineering,
Linux operations, open source, and applied intelligent systems.

Director | Senior Architectural Engineer of Intelligent Software and Systems

PUBLIC ENGINEERING
https://github.com/Llewellynvdm

PROFESSIONAL PROFILE
https://vdm.io/llewellyn

DIRECT CONTACT
https://t.me/llewellynvdm

WINDHOEK / NAMIBIA

Professional Profile

Llewellyn designs platforms, development tools, code-generation systems, reusable frameworks and engineering ecosystems—not only end-user features. His strongest technical centre is PHP and Joomla platform engineering, supported by practical depth in C and C++, Python, Bash, JavaScript and TypeScript, CSS, SQL, APIs, Docker, Linux, networking, OPNsense, self-hosted services, local AI models, Model Context Protocol systems and agent-oriented workflows.

That breadth is integrated rather than incidental. It allows him to move from a domain problem and data model into software architecture, direct implementation, integration, release engineering and the operational environment in which the system must remain secure and maintainable. His judgment is informed by continuing responsibility for applications, servers, networks, databases, services, backups, access control, deployments and continuity.

Much of the public engineering record predates modern generative coding systems. Long-running repositories preserve dated evidence of original, hands-on work and continued support across changing Joomla, PHP, Linux, container and delivery environments. His more recent MCP and applied-AI systems extend that established architecture and automation practice rather than replacing it.

He remains an active engineer while directing company and technical work. He is comfortable operating with substantial autonomy, but also invests deliberately in documentation, code review, internships, mentoring and helping developers progress from implementation toward architectural thinking.

Core Value Proposition

- **Architecture grounded in implementation.** Designs systems and their boundaries, then works directly in the code, data, packaging and deployment needed to make those decisions operational.
- **Platform and ecosystem thinking.** Builds reusable developer capabilities, generators, registries, APIs, deployment systems and release processes that support more than one application or delivery.
- **Production ownership.** Brings practical judgment from long-running responsibility for Linux infrastructure, networking, services, security, backups, maintenance and recovery.
- **Sustained stewardship.** Combines commercial and institutional delivery with dated public histories extending across multiple technology generations: architecture, compatibility work, release maintenance, documentation and community responsibility.
- **Applied intelligent systems.** Integrates AI models, MCP services and agent workflows into established permission, audit, validation and automation practices; AI extends the engineering foundation rather than replacing it.
- **Knowledge transfer.** Teaches architecture, design patterns, system decomposition, infrastructure and responsible AI-assisted development through lectures, public speaking, internships and direct coaching.

Technical Capabilities

Architecture, platforms and application engineering

Software and systems architecture; domain modelling; modular and service-oriented design; developer tooling; generator and compiler-like workflows; reusable frameworks; integration architecture; long-term platform evolution; technical direction.

PHP, Joomla and data systems

PHP application architecture; Joomla extensions and platform tooling; Joomla 3–6 compatibility; component, module and plugin ecosystems; relational modelling and SQL; schema evolution; APIs; data transformation; immutable domain models; contract validation; native-extension integration.

Infrastructure, delivery and operations

Linux administration; Docker and Compose; Bash automation; CI/CD; packaging and release engineering; self-hosted services; deployment; networking; OPNsense; access control; operational continuity; backup and maintenance practice.

Multi-language and intelligent systems

Python libraries and deterministic data pipelines; C/C++ and native PHP extension work; JavaScript/TypeScript browser and service development; CSS and responsive interfaces; MCP integration; local CPU/NVMe inference; AI-agent environments; authenticated and auditable automation.

Leading Public Engineering Evidence

Joomla Component Builder

joomengine/Joomla-Component-Builder
Founder, principal architect, lead developer and long-term maintainer

A multi-generation Joomla extension-development engine and compiler-like build platform supporting Joomla 3 through 6. It brings schema and interface modelling, reusable powers, code generation, packaging and Git-backed distribution into a coherent developer platform.

JCB is the clearest public expression of Llewellyn's principal-level architecture and sustained stewardship. Its public history runs from January 2016 to May 2026, while the project manifest records development from April 2015. It demonstrates platform design, PHP/Joomla depth, generator architecture, compatibility engineering, release ownership and the maintenance of a system used to create and evolve other systems. An identity-aware audit dated 18 July 2026 records 959 attributable commits across its canonical public history; changed-line volume includes delivered generated output and is therefore not presented as manually typed source.

OctoJoom

octoleo/octojoom
Creator, principal Bash engineer and long-term maintainer

A cross-platform Bash utility for deploying and managing Dockerised Joomla and OpenSSH environments through both interactive and direct command-line workflows. Its dated public history runs from May 2021 to February 2026. Current implementation includes approximately 6,573 lines of Bash organised across 175 functions and 71 authored commits.

OctoJoom demonstrates advanced Bash engineering, operator-focused interface design, Docker and Compose operations, Traefik and Portainer integration, OpenSSH, ACME and Cloudflare DNS certificate automation, persistent-volume lifecycle, permissions repair, backup and remote migration. It shows Llewellyn's ability to turn infrastructure knowledge into a practical platform that can be operated consistently beyond his own workstation.

Joomla MCP

joomengine/joomla-mcp
Project architect, principal implementer and maintainer

An embeddable library and self-hosted Model Context Protocol implementation for Joomla administration through bounded, auditable semantic actions. It supports Joomla API and Joomla-native companion paths, guarded writes, principal-bound grants, plans, idempotency, locks, audit events, verification, stdio transport and authenticated Streamable HTTP.

The public `@joomengine/joomla-mcp` 0.7.0 package is supported by typed packaging and OCI, Compose and systemd deployment foundations. Its source-backed catalogue covers all 236 Joomla 6.1 core Web Services route templates across 36 core resource bases. The project connects Llewellyn's mature Joomla architecture with present-day AI and agent systems while retaining explicit production-certification gates.

JoomEngine

octoleo/joomengine
Creator, lead architect and maintainer

The Docker image build system for Joomla Component Builder, expanding Joomla, PHP and runtime matrices into deterministic, versioned images. Its source documents hash-gated discovery, generated manifests, tag leadership and automated publication.

JoomEngine demonstrates container build architecture, reproducibility, registry strategy, Bash-centred build logic and automated release delivery. The generated image contexts are part of the product and release system; they are not represented as hand-authored source volume.

GetBible Librarian

getbible/librarian
Original author, principal architect and maintainer

A Python library for Bible-reference resolution, Scripture retrieval and Unicode-aware search, with bounded, checksum-verified caching, PyPI distribution and separate Query and Search service integration.

Librarian demonstrates Python library architecture, multilingual and Unicode data handling, cache integrity, HTTP API integration, packaging and production hardening. It is part of the broader GetBible platform, where data pipelines, services, applications and distribution channels must agree on durable Scripture contracts.

Official Joomla Docker Images

joomla-docker/docker-joomla
Official Joomla Docker maintainer and senior contributor

The official Joomla Docker image source covers versioned Dockerfiles, entrypoints, PHP configuration, build automation and supported runtime variants. Llewellyn's role is maintained within an official collaborative project rather than claimed as sole ownership.

An identity-aware audit dated 18 July 2026 records 185 attributable commits, with public activity current through 7 July 2026. This work provides independent evidence of sustained container maintenance, compatibility engineering, release responsibility and production distribution within the wider Joomla community.

Current Engineering and Applied AI

Llewellyn's current work extends established architecture, automation and operational practice into intelligent systems, native integration and secure public-service applications.

- **GetBible Robot** — architect, principal implementer and maintainer of a hardened Telegram Scripture service and authenticated Mini App. Current source includes private interaction flows, bounded input handling, persistent translations, responsive and fullscreen reading, accessibility and reduced-motion behaviour, polling and webhook delivery, observability, systemd operation, Docker Compose and published 2.1.0 Linux AMD64 and ARM64 images.
- **GetBible Sword** — C/C++ and Zend/PHP extension integration over the released `getBibleSword` C ABI, exposing streamed PHP APIs without subprocess startup. Engineering evidence includes PIE packaging, PHPT coverage, pinned native dependencies and reproducible golden evidence.
- **GetBible Scripture** — a modern PHP application layer over the native extension, using immutable, lazy Bible-domain objects, contract validation, snapshots, bounded locks, refresh and maintenance state, dependency injection, console commands and Joomla scheduled-task integration.
- **OpenCode Platform** — a version-controlled platform for hardened OpenCode agent virtual machines on Incus. Six Ubuntu 24.04 image variants, manifest-owned policy, capacity-safe builds, credential and network boundaries, reproducible artefacts and controlled Joomla MCP integration demonstrate infrastructure-as-code for agent environments.
- **Colibri Setup** — an operator-focused toolkit for persistent CPU/NVMe-backed local inference while leaving the GPU available. Current capabilities include checksum verification and surgical repair, systemd lifecycle, multi-NVMe reads, CPU-only startup enforcement, Open WebUI integration and adaptive resource profiles capped at 80% of available RAM.
- **GetBible v3 Builder** — a deterministic Python 3.12 pipeline that transforms SWORD modules into API v3 artefacts with token and span annotations, integrity hashes, contract testing and publication automation.
- **OctoLeo engineering suite** — ten authored Octo* repositories covering containerised Joomla operations, release packaging, Composer-package generation, ZIP-to-repository conversion, SHA-512 update metadata, repository synchronisation, Cloudflare security control, mail infrastructure diagnostics, process coordination and safe static infrastructure endpoints. Together they provide substantial current evidence of Bash, networking, security, delivery and operator-focused systems engineering.

Vast Development Method

Founder, Director, Senior Architect and Hands-on Engineer | 2013–present

Vast Development Method is the central professional context for Llewellyn's company leadership, public engineering, commercial delivery and institutional programmes. His responsibility includes company and technical direction, architecture, direct implementation, infrastructure and operations, release and maintenance ownership, open-source work, client continuity, multi-stakeholder delivery and developer development.

GetBible, OctoLeo, JoomEngine and related public ecosystems are company-led, community, product or multi-stakeholder engineering programmes—not separate employers. The company also undertakes shorter and medium-duration consulting engagements across different organisations, grouped by professional objective rather than presented as a fragmented job history.

Company Engineering Estate

git.vdm.dev is Vast Development Method's company-owned Git forge and primary engineering estate. It contains a substantial body of public, private, internal, client and mirrored work, including large-scale proprietary engagements delivered under contractual confidentiality and non-disclosure obligations. The public GitHub organisations are therefore the detailed, shareable part of a larger professional record—not its full extent. Client repositories, system names, architecture, infrastructure, workflows and internal structures are deliberately not itemised. This boundary reflects the trust attached to long-running technical responsibility.

Professional Experience and Responsibility

Software Engineering | 2008–present

Progressed from web and interface delivery into PHP/Joomla application architecture, reusable developer tooling, code generation, APIs, data systems, containers, native integration, platform engineering and intelligent systems. The continuing thread is responsibility for systems that must be delivered, maintained and evolved—not one-off implementation alone.

Linux Systems Administration and Production Operations | 2010–present

Continuous hands-on responsibility for Linux servers, application hosting, databases, Docker services, networking, access control, firewalls, backups, deployment and continuity. This operating experience informs architectural choices around failure modes, maintainability, security boundaries and long-term cost.

Selected Institutional and Long-Term Engagements Delivered Through Vast Development Method

NamPharm | 2016–present

Long-running engineering relationship in Namibia's pharmaceutical-distribution context. Work has included application delivery, continuing maintenance and operational responsibility. The public CV deliberately excludes private system names, detailed architecture, hosting arrangements, integrations and security controls.

East African Community | 2017–present

Institutional systems delivery and continuing operational support for a regional intergovernmental organisation. The engagement demonstrates senior responsibility, continuity and the ability to support information and service delivery across a complex institutional context without exposing private implementation details.

EU–EAC MARKUP | 2019–present

Application and web delivery and maintenance within the EU–EAC Market Access Upgrade Programme context. This is work delivered through Vast Development Method in a multi-stakeholder programme; it is not presented as employment by the participating international institutions.

GIZ

Engineering work in development and social-impact contexts, described at the level of professional objectives, application delivery and support. Client technologies, internal workflows and system details remain confidential.

Broader Consulting Portfolio

Vast Development Method also delivers shorter and medium-duration engagements involving application modernisation, workflow support, integration, data systems, hosting, maintenance and operational continuity. These relationships are grouped professionally rather than listed exhaustively, protecting client confidentiality while reflecting the breadth of company delivery.

Open Source, Leadership and Community

Llewellyn's public responsibility extends beyond code into architecture and strategy, production, operations, testing, compatibility, release maintenance, documentation, mentoring and community service.

- Long-term architecture and stewardship of Joomla Component Builder and its surrounding registries, packages, documentation and distribution systems.
- Official Joomla service across software architecture and strategy, production, operations, automated testing, community web and infrastructure work, Joomla Extensions Directory leadership, official Docker maintenance, security-related service and Afrikaans localisation.
- Google Summer of Code mentoring, project leadership and code review.
- Contribution review, compatibility work, documentation and technical support across public engineering ecosystems.
- Open-source systems that support developer delivery, Scripture publishing, Joomla operation and container distribution over sustained periods.

The breadth of these responsibilities places Llewellyn in environments where engineering changes affect other developers, extension ecosystems and production users, and where compatibility, consensus and release risk matter alongside implementation.

Teaching, Speaking and Mentoring

Developing the next generation of engineers is a sustained professional commitment. Llewellyn combines public technical communication with direct, practical developer development.

- Delivers technical lectures and conference or community presentations on Joomla engineering, algorithms, design patterns, architecture and intelligent-system practice.
- Contributed to the Joomla Forum for the Future in Marbella in 2020, including Technology Stream reporting, and presented Joomla Component Builder at JoomlaDay USA in 2021.
- Creates internship opportunities and supervises practical work, helping junior developers connect implementation tasks to maintainable architecture and production responsibility.
- Coaches developers in design patterns, domain decomposition, modularity, interface boundaries and the progression from writing features to reasoning about systems.
- Provides hands-on guidance in Linux, infrastructure, deployment, operational continuity and the relationship between software design and the environment in which it runs.
- Teaches responsible AI-assisted engineering: using models and agents within clear architectural, validation, permission and review boundaries.
- Produces documentation and public technical instruction intended to make complex systems understandable and maintainable by others.

His communication style is shaped by long practice in public teaching: establishing the underlying model, reducing unnecessary complexity and transferring enough understanding for another person to work independently.

Education and Certification

Certified C++ Programmer

Concentrated Studies in C++ Programming, 2020
C++ Programming I and C++ Programming II were both completed with A grades.

Champlain College

Bachelor of Science in Web Design and Development studies | 2019–2022
Degree not completed.

The supplied institutional audit dated 24 August 2020 records 30 earned institutional credits and a 4.000 institutional and overall GPA, with A grades in every listed completed credit-bearing course. Completed study included networking fundamentals, computer systems, operating systems, relational database design and SQL, project management, critical reading and expository writing, discrete mathematics, and C++ Programming I and II.

Continuing Professional Development

Thirty-four recorded technical and design courses completed from 2011 to 2015 covered PHP and MySQL, object-oriented design, MVC, Java, JavaScript, AJAX, JSON, Node.js, NoSQL, Linux, regular expressions, HTML5, CSS, jQuery, Joomla extension development, C/C++, graphic design and related production tools. Continued learning is also visible in current work spanning TypeScript, native integration, containers, data pipelines, local inference and agent infrastructure.

Mission, Communication and Service

Faith, mission and service are a meaningful long-term foundation in Llewellyn's life. Following his Christian conversion in 1998, he entered full-time mission work in 2000, helped establish the Windhoek church in 2004 and began serving as an elder in 2026.

His ministry includes preparation, teaching, public communication, church responsibility and sustained service. A supplied archive contains 444 distinct audio records from 2021–2025, approximately 249.67 hours, of which 437 are at least twenty minutes. A separate public Brother Llewellyn playlist contains 387 videos, approximately 438.86 hours, with visible dates from 2018–2023. This gives us approximately 831 sermons of approximately 689 hours.

For a professional reader, this record reflects continuity, long-form preparation, public communication, responsibility and service.

Senior Role Fit

Llewellyn is well suited to principal, senior-architect and technical-director work requiring substantial autonomy across software and systems architecture, platform engineering, developer tooling, PHP/Joomla ecosystems, data and integration, Linux operations, container delivery, open-source stewardship, MCP and applied intelligent systems.

He offers the combination of an architect who still implements, a director who understands production detail, and a maintainer prepared to carry systems through release, operation and long-term evolution.