OpenEMR Weekly Meeting Detailed Summary

August 11, 2026

Attendees included: Brady Miller, Stephen Nielson, Stephen Waite, Sherwin Gaddis, David Eschelbacher, Michael Smith (OCE) and Eric Stern (OCE) joining partway through.

Meeting Recording and Summaries

- **Attendees confirmed permission to record meetings** for transcription and summary purposes, in line with common practice across meeting platforms generally.
- **Summaries will omit security specifics.** Published meeting notes will describe decisions and direction without detailing which areas of the codebase are affected or how.
- **Summaries record topics and proposed directions rather than assigning work.** Items noted in these summaries reflect what was discussed and what participants expressed interest in, not commitments or deadlines.
- Recent summaries have been posted to the wiki and referenced in the forums.

Foundation and Governance

- A governance meeting and board meeting were scheduled for the following day.

backup.php — Proposed Removal

Background

- **The script has been a persistent maintenance burden.** The underlying difficulty is that it supports both Windows and Linux through the same code paths, which has made it repeatedly difficult to change safely — addressing one problem has tended to create another.
- Alternative implementation approaches were investigated, but no approach was found that removes the underlying constraint.
- **Assessment: there is no clean solution reachable from the current design.**

What Would Actually Be Lost

- **The backup function has a hard size limit** and stops working above a practical database size, so any growing practice outgrows it regardless. Its value is limited to small, new, or non-technical installations.
- **Log truncation** — moving an oversized log table into a file and starting fresh — is on the same page and does get used. A command-line equivalent was believed to exist.
- **Layout and form export/import** is the one item without clean coverage, and it is of limited practical use: exported layouts are not portable between instances because they carry list and layout options that conflict with the destination database. This is why the project has never hosted downloadable LBF forms. Export is realistically useful only for moving between one's own instances.
- **The EHI export** was raised as an alternative that avoids the file size limitation and is legally mandated to exist. It captures everything holding patient data and produces CSV output, though it does not capture LBF forms and its coverage of globals is uncertain. No importer is built, but the data could be imported by other means.
- **An existing backup and restore utility** written by Rod was located during the meeting at contrib/util/restore — shell scripts covering both backup and restore, handling the sites directory and the database. Its currency is uncertain. Modernizing these as Symfony console commands was discussed, with the counterpoint that a restore utility requiring an installed application is awkward when restoring from scratch.

Position Reached

- **Favored direction: remove it and replace the entry point with a message** explaining the removal and linking to documentation on proper backup practice. If there is community demand, someone can take it over as a module.
- **The recommended practice to document:** a database dump plus a copy of the installation directory, with rsync or similar for ongoing backups. Two steps — and every practice should have a documented backup procedure regardless of which application features exist.
- **Counterargument acknowledged:** small, non-technical practices that do not use the command line and are not active on the forums may lose a capability and simply stop backing up. That is a real risk.
- **Counter to the counter:** a one-click backup enables users to place an unencrypted full copy of their data somewhere it will sit indefinitely. Even a well-built script can enable a poor outcome. A comparison was drawn to phpMyAdmin, previously defended on the same accessibility grounds and later reconsidered.
- Noted that this direction was effectively settled some months ago but was never acted on because other work intervened.

Fax/SMS Module — Core vs. Third-Party

The Testing Problem

- **Changes to the provider integrations cannot be verified end to end without provider accounts.** Code can be reviewed, but the integrations themselves cannot be exercised, so there is no way to confirm that a change preserves functionality.
- **Five channels are present:** Clickatell, SignalWire, Twilio, EtherFax, and RingCentral. Clickatell was previously removed and later restored after a sponsor funded the work.
- **Free account availability is poor.** RingCentral discontinued free developer accounts in 2024; its minimum paid tier was estimated at roughly $40–50 per month. Twilio offers a developer sandbox but exited the fax business two years ago, so it covers SMS only. SignalWire's free tier was unclear during the call. Free eFax services exist but are unlikely to provide a BAA.
- **Accounts currently used for testing are personally held by individual contributors.** Raised as a continuity concern — if these integrations remain in core, the foundation should hold the credentials, and they should be available to automated testing so changes can be regression-tested.
- **Estimated cost:** roughly $100–200 per month for provider accounts across the channels, with a suggested additional allocation for a few hours a month of maintenance. The maintenance budget was considered less likely to be feasible.
- An offer was made to personally cover the account costs, with the view that this should properly sit with the foundation rather than an individual.

The Core-vs-Module Debate

- **Informal policy position stated:** paid vendor integrations do not belong in core. They are better maintained by the individual or the vendor as third-party modules. Core defines the interface and ships either a free provider or a no-op stub, so integration testing can run without incurring costs against a paid service on every commit.
- **Argument for extraction:** a third-party module separates maintenance responsibility cleanly, is easier to maintain independently, and separates funding — money given to core goes to core, money given to the module goes to the module.
- **Argument against:** fax is essential for practices, particularly smaller ones. Roughly 70% of healthcare information in the US still moves by fax. If extracted, a likely outcome is that nobody maintains it and the capability quietly disappears. Prior removals did generate user complaints, and sponsors have paid to have channels reinstated.
- **Counter:** the module is already receiving limited maintenance in core while consuming attention. Sponsors have funded initial integration work but not ongoing upkeep, so the current model has not produced sustained maintenance either.
- **Telemetry was raised as the missing input** — before the foundation funds provider accounts, there should be evidence of actual usage across the segments within the module.
- **Emerging consensus:** keep the framework in core with a defined channel interface, ship one freely testable channel to exercise it, and move the paid vendor channels into separate modules that individuals, vendors, or sponsors can maintain. Described as matching the approach already envisaged for payment integrations.

- **Bundling caveat:** reluctance was expressed about bundling third-party modules inside the distributed package — the position being that the project either takes full ownership and reviews the code, or does not ship it, rather than shipping it while disclaiming responsibility.
- **Extraction would not be trivial:** core contains references to fax/SMS and there are bidirectional dependencies. Some pieces that sat half in core and half in the module were recently consolidated. An audit is needed of what crosses the boundary — what sits in core only because a proper hook is missing, versus what could be cleanly pulled out.
- A side observation: it would be architecturally simpler to support digital faxing only, but many clinics specifically want physical fax machine integration.

Core / Module Coupling — Tracking Issues

Open issues capturing where core is currently coupled to modules, referenced during the discussion:

- **Issue 12222 —** core UI hardcodes oe-module-faxsms paths in the inbox and tabs. https://github.com/openemr/openemr/issues/12222
- **Issue 12221 —** core seeds a WenoExchange background_services row with a non-existent handler path. https://github.com/openemr/openemr/issues/12221
- **Issue 12217 —** decouple core from interface/modules/custom_modules/*. https://github.com/openemr/openemr/issues/12217
- These are the concrete instances of the bidirectional dependency problem — what would have to be resolved before fax/SMS or any similar subsystem could be cleanly extracted from core.

FlySystem and Document Handling

FlySystem documentation: https://flysystem.thephpleague.com/docs/

- **FlySystem is a filesystem abstraction from the League packages.** It replaces direct filesystem calls, adds error handling and reliability, and allows the storage backend to be swapped with a single-line change.
- **Current state: the foundational pieces are in place but little has been migrated onto it.** A large volume of code writes directly to the filesystem throughout the codebase, so full migration is a substantial undertaking, and other priorities have taken precedence.
- **Operational benefit cited:** OCE currently uses GCSFuse to store documents in Google Cloud, which is slow and clunky. With consistent FlySystem use, the backend would simply be Google Cloud Storage with no middleman.
- **Additional benefits:** an abstraction layer allows storage-level guarantees to be applied consistently without the calling code needing to handle them, supports options such as time-limited signed URLs, and resolves Windows versus Linux path handling — which is directly relevant to the backup.php discussion above.
- **The design goal:** a defined set of managed paths where the caller states intent — writing a document, writing a temporary file — and the abstraction places it correctly for the runtime, whether local or remote.
- **The broader principle articulated:** any document entering OpenEMR from outside, whether by fax, CCDA, or another ingestion route, should pass through a consistent pipeline appropriate to its provenance, using the same write API and the existing document class. Patient association varies — a received fax often cannot be attributed to a patient until someone triages it, which is why documents can carry a patient ID of zero and be assigned later in the document viewer. That mechanism already exists, originating from EMR Direct work.
- **Historical context:** an earlier consistency effort found many places bypassing the document class. An issue tracking the remainder has been open for several years and work remains outstanding.
- Observation offered: whether a write goes through the proper facade has depended on the individual developer, and third-party integrations impose their own patterns — SFTP drop folders for EDI being one example.

Module Ecosystem and Upstreaming

- **Sherwin began publishing his module catalog this past week** — releasing everything he has so it is available to anyone who wants it. An automated claims module went out last week and drew a forum enquiry the same day.
- **His stated motivation:** providing a starting point for others. Practices that hire developers often receive work that does not follow OpenEMR's conventions, because the developer is unfamiliar with the platform and works around it rather than with it.
- **His experience on reuse:** the bulk of his work is one-off for a single clinic, and when offered to others the usual response is that they want something different. The clear exception is the prior authorization module, installed over 600 times as of two years ago.
- **The structural problem identified:** practices pay developers for customizations that are never published, so each new customer repeats the same work — and because the result is often not built as a module, it then blocks upgrades. Data was cited supporting the view that a customization one practice wants, a dozen others would also want.
- **The limit acknowledged:** customers paying third-party vendors are under no obligation to publish, and the project cannot maintain a hundred integrations itself. The realistic lever is encouraging publication.
- **Noted as the natural argument for the front controller** — making it easier for everyone to publish modules is the mechanism that makes upstreaming practical.
- **Sinch fax module** was mentioned as an existing OCE module worth reviewing for its architecture, though Sinch is a paid provider with no free tier. Module: https://packagist.org/packages/opencoreemr/oce-module-sinch-fax — provider: https://sinch.com/voice/fax-api/
- **Forum organization:** module discussion currently falls between OpenEMR Development and Third-Party Solutions and Add-Ons, the latter functioning as the advertising area. A subcategory or tag for module authorship was suggested, to give module developers somewhere to find one another and discuss the module system itself. It was observed that module announcements currently appear, draw no replies, and disappear.

Release Status

- **A branch cut was discussed for that day** with release targeted roughly a week out, given scheduling conflicts the following day. Anything can be pulled back after a cut if needed.
- **The release is expected to carry a substantial number of fixes** — enough that the changelog itself demonstrates meaningful progress on the reported backlog.
- **The workflow friction:** ordinary pull request work receives automated review, continuous integration, and full test feedback. Work carried out under embargo receives none of that, which makes it markedly slower and harder. This is what surfaced the fax/SMS testing question.
- **On identifier assignment:** CVE issuance is currently taking around a month industry-wide, so fixes will ship before identifiers are assigned and the identifiers will attach afterward. The view taken: publish the fix and let the administrative process catch up.
- Rolling integration continues and a further cut is expected. Some in-progress features will go into the following week's cut.
- Interest expressed in having additional static analysis tooling report into the GitHub security tab rather than requiring manual runs.

Items Discussed and Proposed

Recorded as topics raised and directions favored, not as assigned work or commitments.

Area	Discussed / Proposed
Board (8/12)	Whether the foundation should fund paid fax/SMS provider accounts to enable automated testing, and whether to fund maintenance time alongside them.
Board (8/12)	If the foundation declines, the alternative is restructuring fax/SMS so core defines the channel interface and vendor channels live in separate modules.
backup.php	Direction favored: remove it and replace the entry point with a notice pointing to documentation on proper backup practice.
Documentation	A wiki page covering recommended backup and restore practice — a database dump plus a copy of the installation directory, with rsync or equivalent for ongoing backups.
Fax/SMS	An audit of what crosses the core/module boundary; there are bidirectional dependencies that would need untangling before extraction is possible.
Fax/SMS	Investigate which providers offer free or sandbox developer accounts suitable for continuous integration.
FlySystem	Broader migration of direct filesystem access onto the abstraction. The foundation is in place; most of the codebase has not yet been migrated.
Forums	A possible subcategory or tag under OpenEMR Development for module authorship, giving module developers a place to find one another.
Release	Branch cut discussed for that day, with release targeted roughly a week out. Rolling integration continues and a further cut is expected.
Telemetry	Raised as the missing input for decisions about removing features — usage data would inform whether the fax module warrants foundation spend.

Recurring Theme

- The same structural question surfaced across every topic — backup.php, fax/SMS, and module publishing alike: what belongs in core, who is responsible for maintaining it, and how it gets tested. The front controller and module architecture work is what makes the preferred answer available in each case.
- Community members willing to take on maintenance of specific modules or subsystems, or to publish customizations they have already built, would have a direct impact on these decisions.