

Blog » De un monorepo a un ecosistema: los paquetes de Derafu que hacen funcionar a LibreDTE Lib Core

De un monorepo a un ecosistema: los paquetes de Derafu que hacen funcionar a LibreDTE Lib Core

Qué paquetes derafu/* usa LibreDTE Lib Core, qué problema resuelve cada uno y cómo pasamos de un solo repositorio a un árbol de dependencias reutilizables.



LibreDTE · 20/10/2026 · #nuevas-funcionalidades

De un monorepo a un ecosistema: los paquetes de Derafu que hacen funcionar a LibreDTE Lib Core

LibreDTE Lib Core es la biblioteca PHP que genera, firma, envía y recibe documentos tributarios electrónicos para el SII. La primera versión es de 2015 y nació como todo lo que nace rápido: **un solo repositorio con todo adentro**. Hoy, en cambio, Lib Core se apoya en 18 paquetes derafu/*, 14 de ellos declarados directamente en su `composer.json`.

En este artículo contamos por qué se hizo esa separación, qué hace cada paquete y cómo se relacionan entre sí.

Cómo era Lib Core en 2015

En la [rama legacy](#) del repositorio se ve la estructura original. En una misma carpeta `lib/` convivían:

- el manejo de XML (`XML.php`),
- la firma electrónica (`FirmaElectronica.php`),

- utilidades de Chile, como el RUT (`Chile.php`),
- los mensajes y traducciones (`I18n.php`),
- la generación de PDF,
- y todo lo relacionado con el SII (`Sii/*`).

Tenía solo dos dependencias externas, `phpseclib` y `TCPDF`, porque lo demás estaba escrito dentro del propio repositorio. Funcionaba, pero tenía problemas técnicos concretos:

- **Acoplamiento al SII y a Chile.** Quien solo necesitaba firmar un XML tenía que instalar una biblioteca de facturación chilena.
- **Poca reutilización.** El código de certificados o de XML servía para otros proyectos, pero estaba atrapado dentro de uno.
- **Pruebas y calidad mezcladas.** Cualquier cambio tocaba un repositorio enorme, y era difícil probar o analizar una pieza por separado.
- **Sin infraestructura común.** Cada parte resolvía por su cuenta los errores, los mensajes y la configuración.

Un ejemplo claro es el token del SII. En la biblioteca original, `getToken()` pedía una semilla, la firmaba y pedía el token **cada vez que se llamaba**. La biblioteca no guardaba nada.

El criterio: sacar lo genérico, dejar lo del SII

La regla fue simple. Lo que no depende del SII ni de Chile se convierte en un paquete propio, más general, que cualquiera puede usar. Lo que sí depende se queda en Lib Core.

Así, Lib Core queda con lo que lo hace único, las reglas de los documentos tributarios y la integración con el SII, y todo lo demás viene de paquetes que se prueban, versionan y analizan por separado. Hoy **todos los repositorios derafu/* que usa Lib Core se analizan con PHPStan en nivel 9 y cobertura de pruebas casi al 100%**.

Los tres que salieron directamente del repositorio original

Tres de las piezas del repositorio de 2015 se convirtieron en paquetes independientes, ya sin ninguna referencia al SII.

`derafu/xml`

Trabaja con documentos XML **como datos**. Convierte XML a arreglos de PHP y de vuelta sin perder elementos, atributos, namespaces ni comentarios. Además valida contra esquemas XSD, consulta con XPath y canonicaliza (C14N), con la codificación que el propio documento declara. Es importante en facturación porque el SII trabaja con XML en ISO-8859-1 y la firma depende de que cada byte esté bien.

derafu/certificate

Carga y lee certificados digitales (X.509 y PKCS#12): extrae el ID (RUN), el nombre, el correo, el emisor y la vigencia, y entrega las claves. Incluye un validador configurable y certificados autofirmados para pruebas. Todo el uso de OpenSSL pasa por un solo servicio, de modo que los errores de una llamada no se mezclan con los de otra.

derafu/signature

Crea y valida firmas digitales, con foco en XML-DSIG, el estándar que usa el SII. Se apoya en los dos paquetes anteriores: `xml` para el documento y `certificate` para las claves. Se puede usar para cualquier firma XML, sin importar el país.

Estos tres son el corazón criptográfico y documental de Lib Core, y son útiles por sí solos en cualquier proyecto PHP que firme XML.

Backbone: la columna vertebral

`derafu/backbone` es el paquete que da **estructura y extensibilidad** a Lib Core. Organiza el código en capas con responsabilidades claras:

1. **Package:** un dominio completo. Lib Core tiene dos, `Billing` y `System`.
2. **Component:** un área funcional dentro del dominio, como `Document`, `Exchange` o `Integration`.
3. **Worker:** quien expone las operaciones y coordina el trabajo.
4. **Job, Handler y Strategy:** la operación atómica, la orquestación y las variantes de implementación.

Todo se descubre con atributos de PHP 8, no con convenciones rígidas de nombres. Por eso agregar una nueva forma de enviar documentos, por ejemplo, es agregar una estrategia, sin tocar el resto. Lib Core tiene cientos de referencias a Backbone, porque es la forma en que está construida.

Backbone da para mucho más: extender la biblioteca, generar APIs automáticas y otras cosas. **Le dedicaremos un artículo propio.**

El resto del ecosistema

Los demás paquetes cumplen funciones de apoyo. Se pueden entender agrupados por rol.

Arranque y datos

- **derafu/kernel:** el núcleo de la aplicación. La clase `Application` de Lib Core extiende su `MicroKernel`, que provee el contenedor de inyección de dependencias, la configuración por ambiente (desarrollo, pruebas, producción) y el caché del contenedor.
- **derafu/config:** clases pequeñas de configuración con acceso anidado, por ejemplo `opciones.sub.clave`.

- **derafu/container**: contenedores de datos (no de servicios), desde “bolsas” simples hasta almacenes validados con JSON Schema. Lib Core los usa en los objetos que intercambian datos entre componentes.
- **derafu/selector**: navegación de estructuras de datos con notación de puntos, sin errores por índices que no existen. Se usa poco en Lib Core y mucho en otros proyectos, pero forma parte de la base común.
- **derafu/repository**: repositorios de datos en memoria. Lib Core los usa para catálogos, como la lista de países de la Aduana.

Estándares y localización

- **derafu/enum**: enumeraciones de uso global.
- **derafu/l10n-cl-enum**: enumeraciones de Chile.
- **derafu/l10n-cl-rut**: validación, formato y cálculo del dígito verificador del RUT y el RUN.

Servicios

- **derafu/cache**: una capa fina sobre PSR-6 y PSR-16. En Lib Core resuelve un problema concreto: el **caché del token del SII**. Antes se pedía un token en cada llamada. Ahora se busca primero en el caché y solo si no existe se solicita uno nuevo, con un TTL por defecto de un minuto. Además, un mecanismo de memoización asegura que el token se calcule una sola vez aunque lleguen varias solicitudes concurrentes.
- **derafu/mail**: envío y recepción de correos, construido sobre Backbone. Lib Core lo usa en el intercambio de documentos, con estrategias para SMTP y para IMAP.
- **derafu/renderer**: una sola interfaz para generar salida desde plantillas, con motores como Twig, Markdown y PDF. Lib Core lo usa para representar los documentos.
- **derafu/twig**: el entorno de Twig con sus extensiones, del que depende **renderer**.

Base común

- **derafu/support**: utilidades esenciales para fechas, textos, arreglos, hidratación de objetos y fábricas.
- **derafu/translation**: mensajes traducibles. Casi todos los paquetes lo usan, por eso **las excepciones de los paquetes tienen mensajes que se pueden traducir**, incluyendo su traducción al español.

El árbol de dependencias

El siguiente diagrama muestra cómo se relacionan los paquetes. Los de arriba son los que Lib Core usa directamente. Los de abajo son la base que comparten todos.

Hay dos cosas que vale la pena notar:

- `derafu/translation` y `derafu/enum` no dependen de ningún otro paquete `derafu/*`. Son las hojas del árbol.
- `derafu/signature` es el que más dependencias `derafu/*` reúne entre los de dominio, porque une XML, certificados y utilidades.

Qué ganamos

- **Reutilización.** XML, certificados y firma se pueden usar sin llevarse facturación chilena.
- **Calidad homogénea.** Cada paquete tiene sus propias pruebas y su análisis estático en nivel 9 con cobertura de pruebas casi al 100%.
- **Cambios acotados.** Se corrige o mejora una pieza sin tocar el resto.
- **Errores claros.** Excepciones traducibles con el detalle de lo que falló.

- **Rendimiento.** El token del SII se reutiliza en vez de pedirse en cada llamada.
- **Extensibilidad.** Gracias a Backbone, agregar variantes (otra forma de enviar, otro formato de salida) no obliga a modificar el núcleo.

Lo que viene

Esta separación abre varios temas que se pueden abordar en próximos artículos: Backbone a fondo, las APIs automáticas que se obtienen con él, el uso de los paquetes de firma y XML fuera de Lib Core, y más. **Síguenos para no perderte los siguientes.**

La documentación de cada paquete está en derafu.dev, y el código de LibreDTE Lib Core en [GitHub](https://github.com).

Última actualización el 20/10/2026