Modularisation de la suite BY+

Version courte · septembre 2026

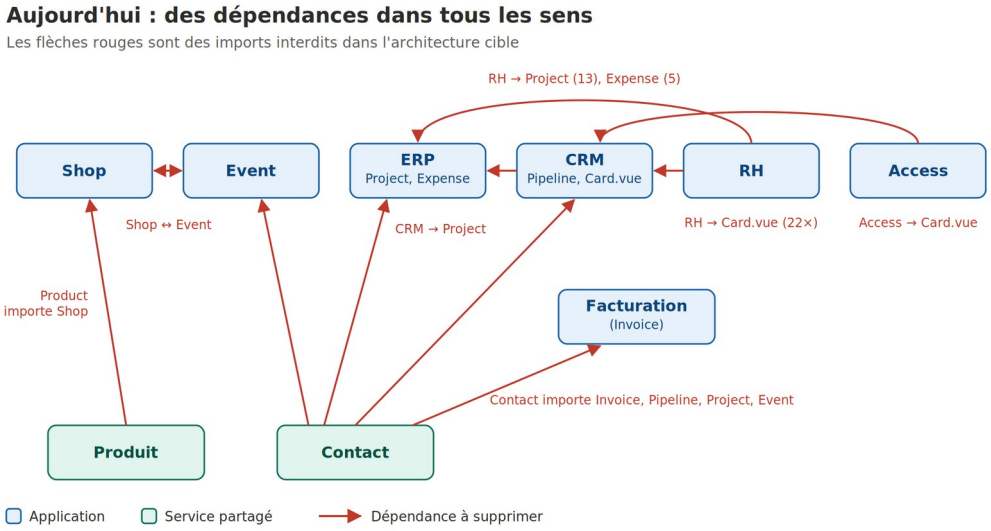
> **En une phrase**
> Une seule application découpée en **6 apps activables par client** (Shop, ERP, CRM, RH, Access, Event), rangées en **3 couches** et en **packages**, progressivement, sans microservices ni réécriture.

Objectifs

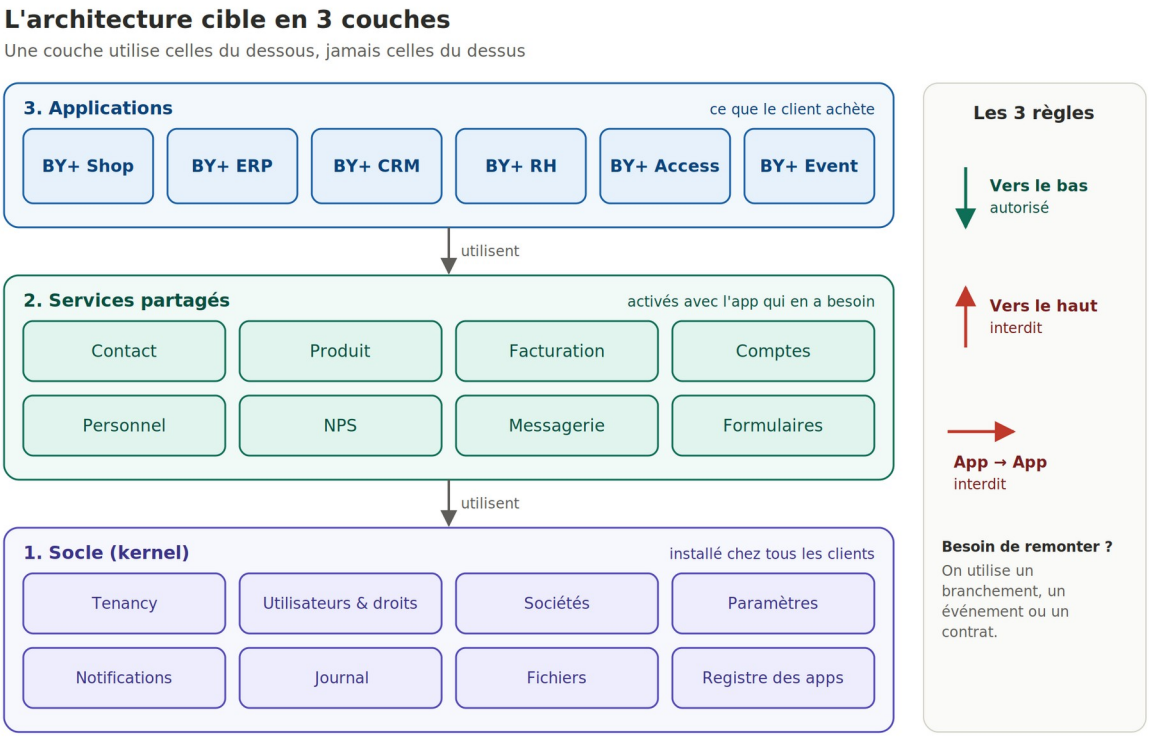
- **Vendre à la carte** : un client ne charge que ses apps.
- **Apps qui coopèrent** sans se connaître (ex. factures visibles dans la fiche contact).
- **Modifier sans peur** : toucher Event ne casse pas RH.

1. Le problème

57 modules où tout dépend de tout : on ne peut rien désactiver.



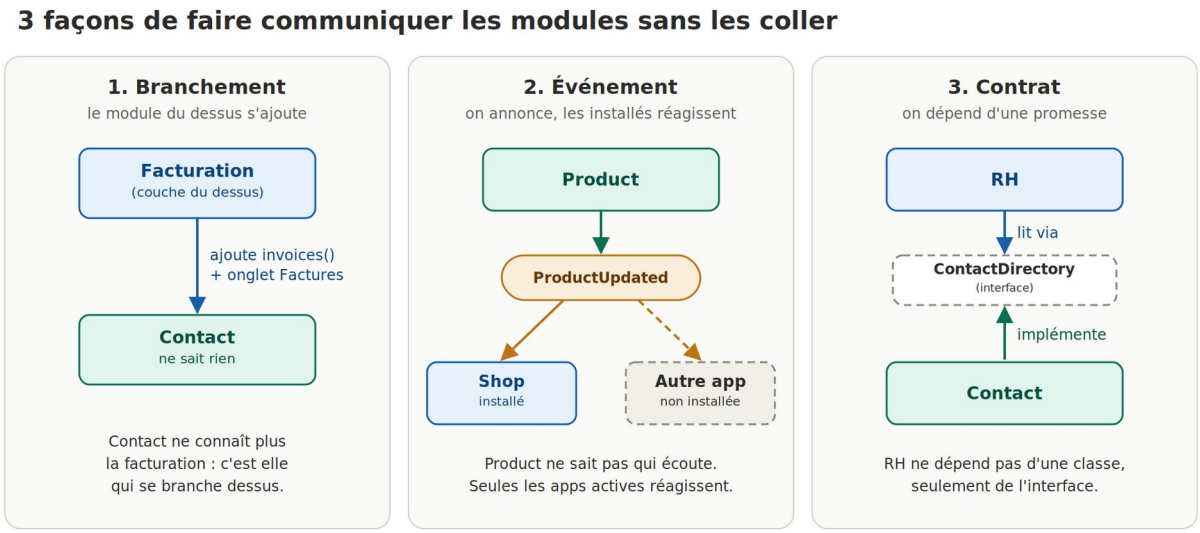
2. L'architecture cible



Service partagé	Shop	ERP	CRM	RH	Access	Event
Contact	✓	✓	✓	✓	✓	✓
Produit	✓	✓	✓			
Facturation		simple	complète	?		
Personnel		✓		✓		
NPS	✓	✓	✓			✓
Messagerie		✓				✓
Formulaires			✓			✓

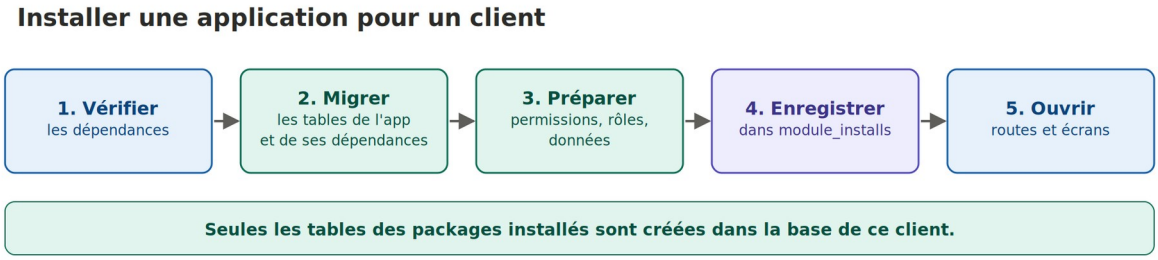
3. Découper sans coller

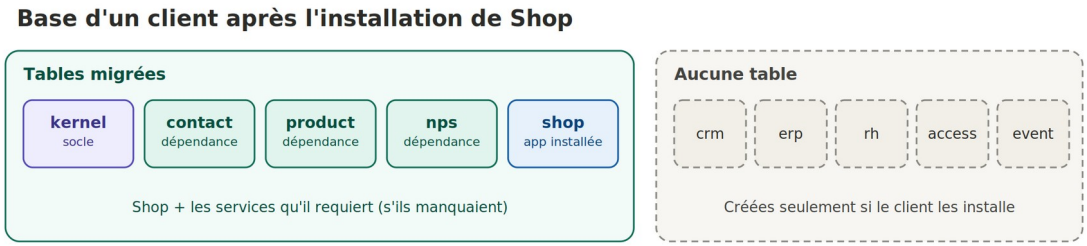
Chaque module devient un **package** (dossier autonome dans la même application), et chaque import interdit est remplacé par l'un de ces trois mécanismes :



4. Installer une application

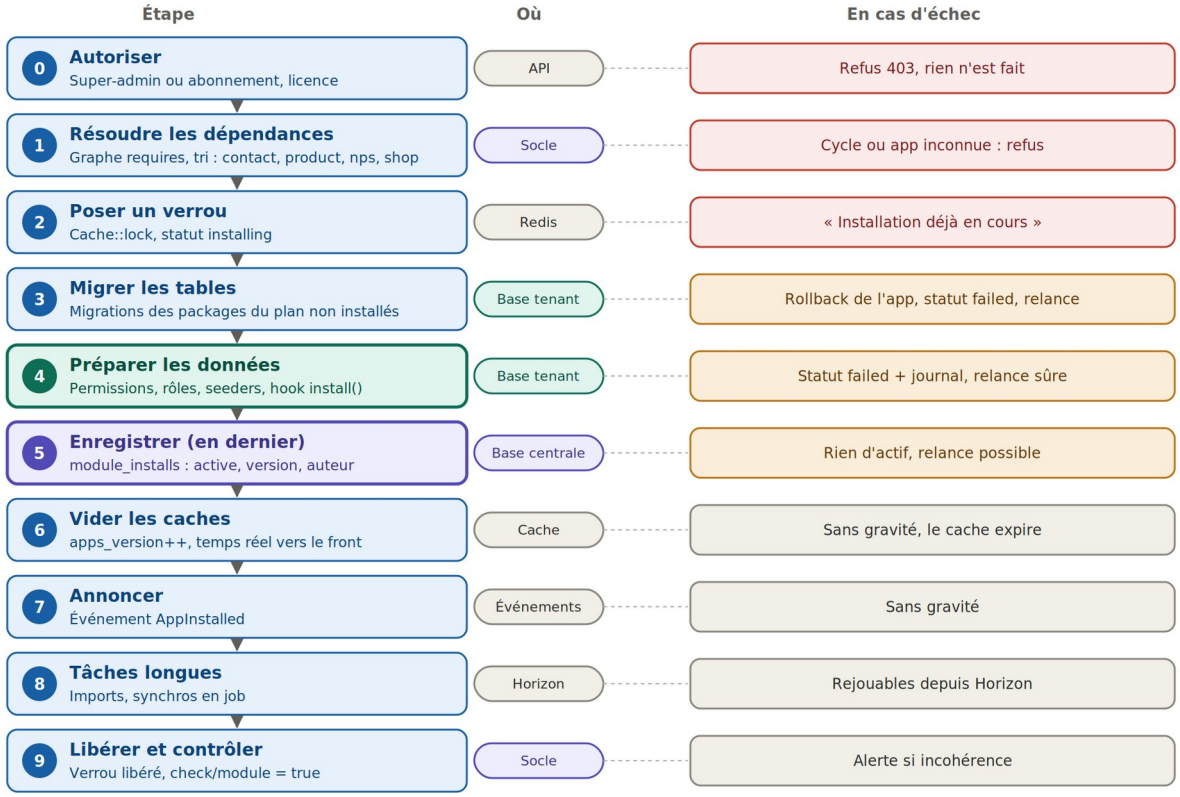
On ne migre **que les tables de l'app installée**, plus celles de ses dépendances si le client ne les a pas encore. Les autres apps n'ont aucune table chez ce client.





5. L'installateur, étape par étape

AppInstaller : les 10 étapes, et ce qui se passe en cas d'échec



Initialisation des données (étape 4)

Dans la base du client, pour chaque package dans l'ordre des dépendances : **migrations du package (étape 3), puis permissions → rôles → seeders → hook install()** (entrepôt par défaut, étapes de pipeline, numérotation des factures).

Tout est rejouable sans doublon. L'app n'est marquée **active** dans la base centrale qu'**après** (étape 5).

Règles liées aux migrations par app

Chaque package garde ses migrations dans son propre dossier et son propre suivi.

Mise à jour : une commande `byplus:migrate` migre, client par client, seulement les packages qu'il a installés (remplace `tenants:migrate` global).

Clés étrangères : uniquement vers les tables d'un package listé dans `requires`.

Nouveau client : seul le socle est migré, puis ses apps par défaut via l'installateur.

Lancer la migration de l'app : AppMigrator

La migration tourne **dans la base du client** et se limite **au dossier du package**.

```
// Socle — app/Core/AppMigrator.php
class AppMigrator
{
    public function migrate(Tenant $tenant, string $app): void
    {
        $path = base_path("packages/byplus/app-{$app}/database/migrations");

        $tenant->run(function () use ($path) {
            Artisan::call('migrate', [
                '--path'     => $path,  // seulement les tables de cette app
                '--realpath' => true,
                '--force'    => true,   // obligatoire en production
            ]);
        });
    }
}
```

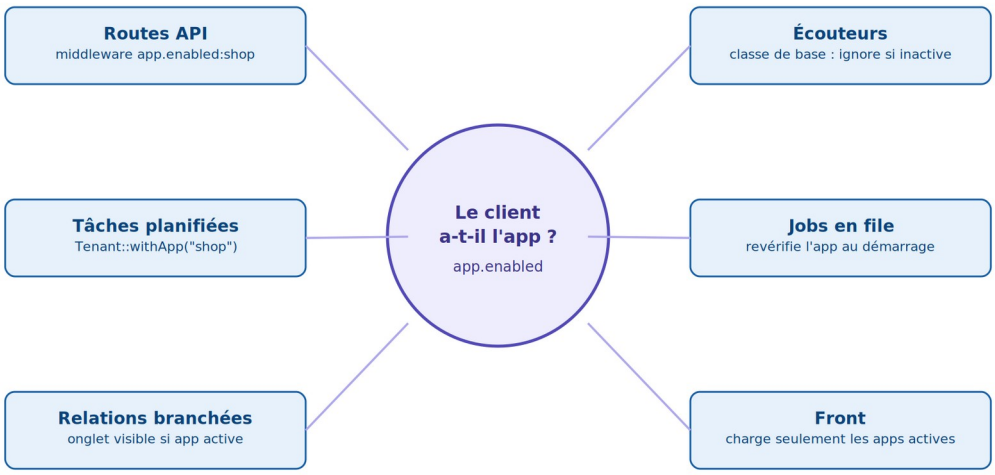
Appel dans l'installateur, pour chaque package manquant :

```
foreach ($plan->missingFor($tenant) as $package) {
    $this->migrator->migrate($tenant, $package->key);    // 1. tables de l'app
    $tenant->run(fn () => $package->install());          // 2. données par défaut
    $this->registry->markActive($tenant, $package, $by); // 3. activation
}
```

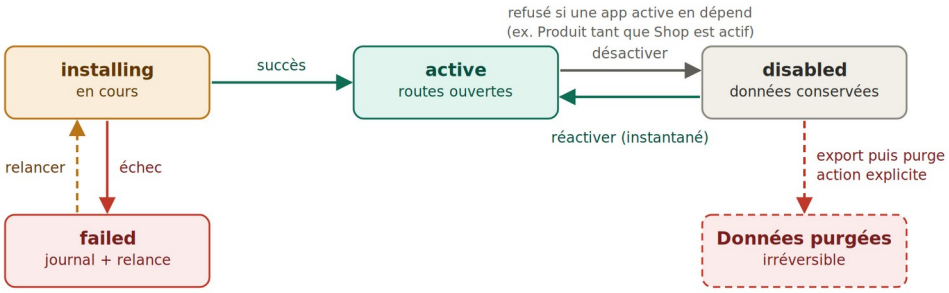
- **Pas de loadMigrationsFrom** dans le provider du package : sinon un migrate global créerait les tables chez tous les clients.
- **Relançable** : la table `migrations` du client garde la trace ; seules les nouvelles migrations passent (utile aussi pour les mises à jour).

6. Protéger et gérer

6 points d'entrée à protéger : le code de toutes les apps tourne chez tous les clients



Cycle de vie d'une application chez un client



Travaux à réaliser

- ☐ Séparer l'endpoint en `install` et `disable`.
- ☐ Ranger les migrations par package ; retirer la migration globale de `MigrateDatabase`.
- ☐ Commande `byplus:migrate` : migrations par client, seulement pour ses packages installés.
- ☐ Créer **AppRegistry** et **AppInstaller**.
- ☐ Brancher et rendre idempotents les seeders existants.
- ☐ Middleware `app.enabled` sur toutes les routes (commencer par Access).
- ☐ Filtrer écouteurs, tâches planifiées et jobs par app active.
- ☐ Tests par app : routes de l'app en 200, des autres en 403.