



Rapport de stage

Migration du système
SAPIENS : Transition de
Drupal 7 vers Drupal 10 à
l'Université de Montpellier

Réalisé par

Raphaël LAMBERT

Maître de stage

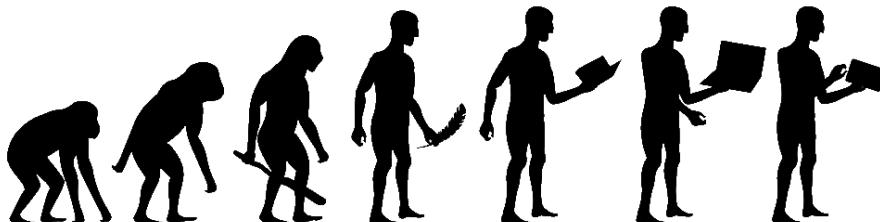
Frédéric GUITTON

Tuteur de stage

Antoine CHOLLET

Pour l'obtention du BUT 2

Année universitaire 2024-2025



S.A.P.I.E.N.S

Salles Pédagogiques Informatiques d'ENSEignements

Remerciements

Je tiens à exprimer ma profonde gratitude envers toute l'équipe de la DSIN qui m'a accueilli à bras ouverts pendant toute la durée de mon stage. Leur disponibilité, leur bienveillance et leur professionnalisme ont créé un environnement propice à mon apprentissage et à mon développement professionnel.

Mes remerciements s'adressent particulièrement à Monsieur Frédéric Guitton, qui a joué le double rôle de maître de stage et de client. Sa disponibilité constante, ses conseils avisés et sa patience m'ont permis de mener à bien ce projet complexe. Son expertise technique et sa vision claire des objectifs ont été des guides précieux tout au long de cette expérience.

Je souhaite également remercier Monsieur Cyril Dorcy, chef du service, qui m'a accueilli au sein de son équipe et m'a accordé sa confiance pour participer à ce projet stratégique pour l'université.

Ma reconnaissance va aussi à Monsieur Loïc Bonavent, sans qui cette opportunité n'aurait pas été possible. Sa recommandation m'a ouvert les portes de la DSIN et je lui en suis profondément reconnaissant.

Enfin, je remercie Monsieur Antoine Chollet, mon tuteur de stage, qui, bien que nos échanges aient été moins fréquents, a toujours été présent et disponible en cas de besoin.

Cette expérience enrichissante n'aurait pas été aussi formatrice sans la collaboration et le soutien de chacune de ces personnes, ainsi que de tous les membres de la DSIN avec qui j'ai eu l'occasion de travailler.

Résumé

Ce rapport présente la migration* du système SAPIENS* (SAlles Pédagogiques Informatiques d'ENSEignements) de l'Université de Montpellier, de Drupal* 7 vers Drupal 10. Ce projet, réalisé au sein de la Direction du Système d'Information et du Numérique (DSIN)*, visait à moderniser une application critique pour la gestion des ressources pédagogiques informatiques.

La migration^[21] a impliqué une refonte complète de l'architecture, notamment la réécriture des modules personnalisés et l'optimisation des intégrations avec les services externes comme Ivanti*^[131]. Une attention particulière a été portée à la restructuration du système de gestion des services (MySQL*, PostgreSQL*, Web, Oracle*) en remplaçant les field_collections^[16] par des Paragraphs^[6], tout en maintenant la compatibilité avec les processus existants.

L'implémentation d'un système de cache^[111] innovant et l'optimisation des performances de recherche ont permis d'améliorer significativement l'expérience utilisateur. Le développement s'est appuyé sur une méthodologie agile* adaptée, utilisant Tuleap^[91] pour la gestion de projet et assurant une documentation exhaustive* sur le WikiDSIN*.

Le projet a abouti à une architecture moderne, sécurisée et performante, préparant SAPIENS pour les évolutions futures tout en garantissant la continuité des services pour la communauté universitaire.

Mots-clés

Migration système, Drupal 10, PHP*, Symfony*, Développement web, Gestion de projet agile, Intégration continue*, Architecture logicielle, Performance applicative, Système d'information universitaire

Glossaire

AJAX : Ajax est une technique de programmation qui permet une communication asynchrone entre le client (navigateur) et le serveur

Algorithme de Levenshtein : Formule mathématique qui mesure la similarité entre deux mots, utilisée ici pour améliorer la recherche même quand il y a des fautes de frappe.

API (Application Programming Interface) : Interface qui permet à différents logiciels de communiquer entre eux, comme un "langage commun" permettant l'échange d'informations.

Apogée : Système d'information de gestion de la scolarité utilisé par les universités françaises pour gérer les inscriptions et parcours des étudiants.

Apprentissage critique : Ensemble de compétences spécifiques à acquérir pendant une formation, qui participent au développement d'une compétence plus large.

Annuaire web : ce système permettant aux enseignants de retrouver les espaces web de leurs étudiants.

Architecture modulaire : Organisation du code en "briques" indépendantes qui peuvent être modifiées sans affecter le reste du système, comme des pièces de Lego qu'on peut ajouter ou retirer facilement.

Architecture monolithique : Application construite en un seul bloc, contrairement à une architecture modulaire.

Asynchrone : Se dit d'une opération qui continue à s'exécuter en arrière-plan pendant que le reste du programme continue à fonctionner.

Audit technique : Examen détaillé du système pour identifier ses forces et faiblesses.

Burndown chart : Graphique qui montre l'avancement du travail au fil du temps, comme un compteur qui descend jusqu'à zéro.

Cache : Système de stockage temporaire qui garde en mémoire des informations fréquemment utilisées pour y accéder plus rapidement, comme un "pense-bête" informatique.

CAS (Central Authentication Service) : Système qui permet aux utilisateurs de se connecter une seule fois pour accéder à plusieurs services de l'université avec les mêmes identifiants.

Cérémonies Scrum : Réunions régulières (daily scrum, rétrospective, etc.) qui permettent à l'équipe de s'organiser et de s'améliorer.

Champs computed : Champs dont la valeur est calculée automatiquement à partir d'autres données.

Commits claire : Pratique de développement qui consiste à effectuer des enregistrements de modifications petits et focalisés sur une seule tâche ou correction à la fois.

Conteneurisation : Technique qui permet d'emballer une application avec tout ce dont elle a besoin pour fonctionner, comme une boîte autonome.

Cron : Système qui permet d'exécuter automatiquement des tâches à des moments précis.

CSITools : Utilisé en qualité de WebService, outil spécifique mentionné pour la récupération des données.

Daily scrums : Courtes réunions quotidiennes où l'équipe fait le point sur l'avancement du travail.

DDev : Outil qui crée un environnement de développement local standardisé, permettant à tous les développeurs de travailler dans les mêmes conditions.

Déploiement : Processus de mise en place d'un système sur les serveurs où il sera utilisé.

Dette technique : Compromis techniques faits dans l'urgence qu'il faudra corriger plus tard, comme des "raccourcis" qui finissent par poser problème.

Documentation exhaustive : Ensemble complet des informations qui expliquent comment le système fonctionne et comment l'utiliser.

DSIN (Direction du Système d'Information et du Numérique) : Service informatique central de l'université qui gère tous les systèmes et outils numériques.

Drupal : Système de gestion de contenu web (CMS) qui permet de créer et gérer des sites internet. La migration concerne le passage de la version 7 à la version 10.

Environnement de développement : Ensemble des outils et logiciels nécessaires pour créer et tester l'application, comme l'atelier d'un artisan.

Field Collections : Ancien système de Drupal 7 permettant de regrouper plusieurs champs d'information ensemble, comme une "boîte" contenant différentes données.

Geofield : Module Drupal qui gère les données géographiques, permettant d'afficher et manipuler des informations de localisation sur une carte.

GIT : Git est un logiciel de gestion de versions. Il permet de suivre les modifications de votre code et d'organiser vos projets de développement. C'est un outil essentiel, que vous travaillez seul, en équipe

Hook : Fonction spéciale dans Drupal qui permet d'intervenir à des moments précis du fonctionnement du système, comme un "point d'accroche" pour ajouter des fonctionnalités.

Intégration continue : Pratique qui consiste à vérifier automatiquement que les nouveaux ajouts au code fonctionnent bien avec l'existant.

Ivanti : Logiciel externe utilisé pour gérer l'inventaire du matériel informatique de l'université.

JSON : Format de données qui permet d'organiser et échanger des informations entre différents systèmes informatiques de manière structurée et lisible.

Journalisation : Enregistrement automatique des événements qui se produisent dans le système, comme un journal de bord.

Knowledge management : Gestion des connaissances, façon d'organiser et de partager les informations au sein d'une organisation.

LDAP : Lightweight Directory Access Protocol, protocole permettant l'accès à un service d'annuaire.

Mapping dynamique : Système qui fait automatiquement correspondre des données entre différents formats ou systèmes.

Méthodologie agile : Approche de gestion de projet flexible qui s'adapte aux changements et privilégie la collaboration et les retours réguliers.

Migration : Processus de transfert des données et fonctionnalités d'un ancien système vers un nouveau, comme un déménagement numérique.

Mind map : Représentation visuelle des idées et de leurs connexions, comme une carte des concepts.

MySQL/PostgreSQL/Oracle : Différents types de bases de données qui stockent et organisent les informations du système.

Normalisation des données : Processus qui consiste à organiser les données de manière cohérente et standardisée.

Paragraphs : Nouveau système de Drupal 10 remplaçant les Field Collections, offrant une façon plus flexible d'organiser et présenter le contenu.

PHP : Langage de programmation utilisé pour créer les fonctionnalités du site web.

Processus métier : Les différentes étapes et règles qui définissent comment une organisation fonctionne au quotidien.

Product Backlog : Liste de toutes les fonctionnalités à développer, comme une liste de courses pour le projet.

Product Owner : Personne responsable de définir ce que doit faire le produit et de prioriser les tâches.

Référence d'entité : Concept Drupal un contenu qui référence d'autres contenus.

Requête Ivanti : Demande d'information envoyée au système Ivanti pour obtenir des données sur le matériel informatique.

Rétrospective : Réunion où l'équipe discute de ce qui a bien ou mal fonctionné pour s'améliorer.

SAPIENS (SAlles Pédagogiques Informatiques d'ENSEignements) : Application qui gère les ressources informatiques pédagogiques de l'université.

SEP (Service Environnements & Production) : Équipe au sein de la DSIN qui s'occupe de la gestion des environnements informatiques.

Sprint : Période de travail fixe pendant laquelle l'équipe doit réaliser certains objectifs définis.

"Start, Stop, Continue" : Format de réunion où l'on décide ce qu'il faut commencer, arrêter ou continuer à faire.

SVG : Format d'image qui permet de créer des graphiques qui restent nets quelle que soit leur taille.

Symfony : Framework (cadre de développement) PHP moderne utilisé par Drupal 10 pour construire des applications web de manière structurée.

Synchronisation bi-hebdomadaire : Mise à jour des informations qui se fait deux fois par semaine.

Système wrapper : Code qui sert d'intermédiaire entre deux systèmes pour les faire communiquer plus facilement.

Tuleap : Outil de gestion de projet utilisé pour organiser et suivre l'avancement du travail de migration.

Twig : Twig est un moteur de templates pour le langage de programmation PHP, utilisé par défaut par le framework Symfony

User Story : Description simple d'une fonctionnalité du point de vue de l'utilisateur.

Vélocité : Mesure de la quantité de travail que l'équipe peut accomplir dans un sprint.

Webservice : Service qui permet à différentes applications de communiquer via Internet, comme un "pont" entre différents systèmes.

WikiDSIN : Plateforme interne de documentation où sont stockées toutes les informations techniques de la DSIN, basé sur le logiciel Confluence.

Workflow : aussi appelé flux de travaux ou flux opérationnel, désigne une suite de tâches ou d'opérations qui doivent être réalisées par un individu ou un groupe d'individus selon un ordre spécifique

YAML : Format de fichier simple utilisé pour configurer les applications, plus facile à lire et écrire que d'autres formats techniques.

Bibliographie

- [1] « Drupal 10 Documentation », Drupal.org, écrit par Jess (xjm), le 15 Dec 2022, mis à jour 13 Jan 2023. [En ligne]. Disponible sur : <https://www.drupal.org/project/drupal/releases/10.0.0> . [Consulté le : 13-janv-2025].
- [2] « Migration de Drupal 7 vers Drupal 10 : Votre guide pas à pas », evolvingweb, par Andrew Sus le nov 9, 2023. [En ligne]. Disponible sur : <https://evolvingweb.com/fr/blog/migration-de-drupal-7-vers-drupal-10-votre-guide-pas-pas> . [Consulté le : 14-janv-2025].
- [3] Diapositive de formation drupal déploiement et industrialisation, Trained People, 2023. [Documentation interne].
- [4] « Symfony 6.4 Documentation », Symfony.com. [En ligne]. Disponible sur : <https://symfony.com/doc/6.4/index.html> . [Consulté le : 24-janv-2025].
- [5] DropTeam, « Rapport d'audit - Migration SAPIENS Drupal 7 vers Drupal 10 », Écrit par Romain DORTIER, Romain JARRAUD, le 19/09/2024 [Consulté le : 14-janv-2025].
https://drive.google.com/file/d/1uQN-RBR8n0jGI8ZSTbnSrXNWB_NCdQqF/view?usp=drive_link .
- [6] « Drupal Paragraphs Module Documentation », Drupal.org. [En ligne]. Disponible sur : <https://www.drupal.org/docs/contributed-modules/paragraphs> . [Consulté le : 04 fév 2025].
- [7] « Geofield », Drupal.org, Crée par Patrick Hayes le 9 March 2011, mis à jour 6 November 2024. [En ligne]. Disponible sur : <https://www.drupal.org/project/geofield> . [Consulté le : 15-janv-2025].
- [8] « Drupal Migrate API Documentation », Drupal.org. [En ligne]. Disponible sur : <https://www.drupal.org/docs/drupal-apis/migrate-api> . [Consulté le : 17 janv 2025].
- [9] « Tuleap Agile Documentation », Tuleap.org. [En ligne]. Disponible sur : <https://docs.tuleap.org/user-guide/intro.html> . [Consulté le : 24 janv 2025].
- [10] « Documentation CAS Drupal », Drupal.org, Crée par David Metzler le 16 Aout 2006, mis à jour le 9 Janvier 2025. <https://www.drupal.org/project/cas>.

[11] « 12.1. Concept : cache », Drupal.org, Écrit par Jennifer Hodgdon mis à jour 28 août 2024. [En ligne]. Disponible sur : https://www.drupal.org/fr/docs/user_guide/fr/prevent-cache.html . [Consulté le : 29-janv-2025].

[12] « Symfony Security Best Practices », Symfony.com. [En ligne]. Disponible sur : <https://symfony.com/doc/6.4/security.html> . [Consulté le : 23-janv-2025].

[13] “Ivanti”, Ivanti, écrit le 12 juillet 2018 [En ligne]. Disponible sur : <https://www.ivanti.com/fr/company/press-releases/2018/ivanti-enhances-user-experience-and-desktop-control> [Consulté le : 20-janv-2025].

[14] “Understanding hooks”, Drupal.org, dernière mise à jour 21 Sept. 2023, [En ligne]. Disponible sur : <https://www.drupal.org/docs/develop/creating-modules/understanding-hooks>

[15] “Get Started with DDEV”, DDev, dernière mise à jour 27 Decembre 2024, [En ligne]. Disponible sur : <https://ddev.readthedocs.io/en/stable/>

[16] “Field Collection”, Drupal.org, créée par Wolfgang Ziegler le 20 Octobre 2010, [En ligne]. Disponible sur : https://www.drupal.org/project/field_collection

[17] “Rules Module”, Drupal.org, créé par Wolfgang Ziegler le 7 November 2007, [En ligne]. Disponible sur : <https://www.drupal.org/project/rules>

Sommaire

Introduction.....	1
I - Contexte du stage.....	2
1.1 - Présentation de l'Université de Montpellier.....	2
1.2 - Présentation de la DSIN et du SEP.....	2
1.3 - Le Projet SAPIENS : Un Enjeu Stratégique.....	3
II - Analyse technique.....	4
2.1 - État des lieux de l'infrastructure existante.....	4
2.2 - Analyse critique et points d'attention.....	5
2.3 - Stratégie de migration.....	6
III - Réalisation technique.....	7
3.1 - Établissement de l'environnement de développement.....	7
3.2 - Migration des structures de données fondamentales.....	8
3.2.1 - Architecture des contenus géographiques.....	8
3.2.2 - Développement du module de gestion des salles.....	9
3.2.3 - Système de migration automatisée.....	10
3.2.4 - Système de gestion des logiciels et optimisation des performances.....	11
3.2.5 - Développement de l'annuaire des espaces web.....	13
3.2.6 - Modernisation du module de gestion des examens.....	16
3.3 - Intégration avec l'infrastructure universitaire.....	18
3.4 - Déploiement et documentation.....	21
3.5 - Développement post-migration : système de suivi des garanties matérielles.....	21
IV - Méthodologie et gestion de projet.....	24
4.1 - Cadre méthodologique et adaptation de Scrum.....	24
4.2 - Management visuel et pilotage du projet.....	25
4.3 - Système de gestion de version et intégration continue.....	25
4.4 - Collaboration inter-équipes et gestion des dépendances.....	27
4.5 - Démarche qualité et amélioration continue.....	28
4.6 - Documentation et knowledge management*.....	29
V - Perspectives et améliorations futures.....	29
5.1 - Intégration native de la procédure d'examen.....	30
5.2 - Vers une autonomisation des enseignants.....	30
5.3 - Architecture technique envisagée.....	31
Conclusion.....	32
Annexes.....	34

Table des figures

Figure 1 : Logo SEP	2
Figure 2 : Organigramme DSIN	3
Figure 3 : Structure Contenu Base	8
Figure 4 : Exemple Contenu Campus	8
Figure 5 : Exemple contenu salle	10
Figure 6 : Vue Liste de logiciel par salle	12
Figure 7 : Page synchronisation LDAP	14
Figure 8 : Page synchronisation UE	15
Figure 9 : Page Annuaire des espaces Web	16
Figure 10 : Exemple Bloc Examen	17
Figure 11 : Architecture de communication service	19
Figure 12 : Bloc affichage service	20
Figure 13 : Page Garantie des équipements	22
Figure 14 : Outil de suivi Tuleap	25
Figure 15 : Environnement GIT	26
Figure 15 : Commit GIT	27

Introduction

Dans un contexte où la transformation numérique bouleverse profondément le paysage de l'enseignement supérieur, les établissements universitaires font face à des défis technologiques majeurs. L'Université de Montpellier, consciente de ces enjeux, s'engage dans une modernisation continue de son infrastructure numérique pour répondre aux besoins croissants de sa communauté. Au cœur de cette transformation se trouve le système SAPIENS* (SAlles Pédagogiques Informatiques d'ENSEignements), un outil stratégique pour la gestion des ressources pédagogiques numériques.

La Direction du Système d'Information et du Numérique (DSIN) a identifié la nécessité critique de faire évoluer la plateforme SAPIENS, actuellement basée sur Drupal* 7, vers une architecture moderne utilisant Drupal 10 [\[11\]](#). Cette migration [\[21\]](#) représente bien plus qu'une simple mise à jour technique : elle constitue une opportunité de repenser l'architecture applicative pour améliorer la maintenabilité, la sécurité [\[12\]](#) et l'expérience utilisateur. La fin annoncée du support de Drupal 7 rend cette transition non seulement stratégique mais impérative pour garantir la pérennité du système.

Mon stage s'inscrit dans cette dynamique de modernisation, avec pour mission principale la participation active à cette migration complexe. Les objectifs sont multiples : assurer une transition fluide des fonctionnalités existantes, optimiser les performances du système, et introduire de nouvelles capacités techniques tout en préservant l'intégrité des données et des processus métier*.

Ce rapport présente la démarche adoptée pour relever ces défis, en commençant par une analyse approfondie du contexte et des besoins, suivie d'une description détaillée des solutions techniques mises en œuvre. Une attention particulière est portée à la méthodologie de développement et aux choix architecturaux qui ont guidé ce projet de modernisation.

La complexité de cette migration réside notamment dans la nécessité de maintenir la compatibilité avec les systèmes externes comme Ivanti*[\[13\]](#), tout en modernisant l'architecture pour exploiter les nouvelles possibilités offertes par Drupal 10 et le framework Symfony[\[4\]](#). Ce travail exige une compréhension approfondie non seulement des aspects techniques, mais également des processus métier et des besoins des utilisateurs.

L'expérience acquise au sein de la DSIN m'a permis de développer une vision globale des enjeux de la transformation numérique dans le contexte

universitaire, tout en approfondissant mes compétences techniques en développement web et en gestion de projet informatique.

I - Contexte du stage

1.1 - Présentation de l'Université de Montpellier

L'Université de Montpellier est une institution publique d'enseignement supérieur et de recherche qui regroupe plusieurs campus. Elle propose une large variété de formations et s'appuie sur une infrastructure numérique stratégique pour assurer ses activités. Elle dispose d'une DSIN (Direction du Système d'Information et du Numérique) performante qui assure le bon fonctionnement et l'évolution de ses services numériques.

1.2 - Présentation de la DSIN et du SEP

La Direction du Système d'Information et du Numérique (DSIN) joue un rôle central dans le développement, la maintenance et l'évolution des outils numériques universitaires.

Au sein de la DSIN, le Service Environnements & Production (SEP)* sous la direction de C. Dorcy, le service adopte une approche agile et cross-fonctionnelle, structurée autour de deux équipes complémentaires. Cette organisation permet une grande flexibilité dans la gestion des projets tout en maintenant un haut niveau d'expertise technique.



Figure 1 : Logo SEP

La coordination des équipes s'appuie sur une répartition claire des responsabilités :

- **Équipe Environnements Applicatifs** : administration des systèmes serveurs, bases de données, et support technique.
- **Équipe Environnements Utilisateurs** (sous la coordination de M. Frederik Guitton) : gestion des systèmes clients et serveurs, déploiement* des applications et support utilisateur.

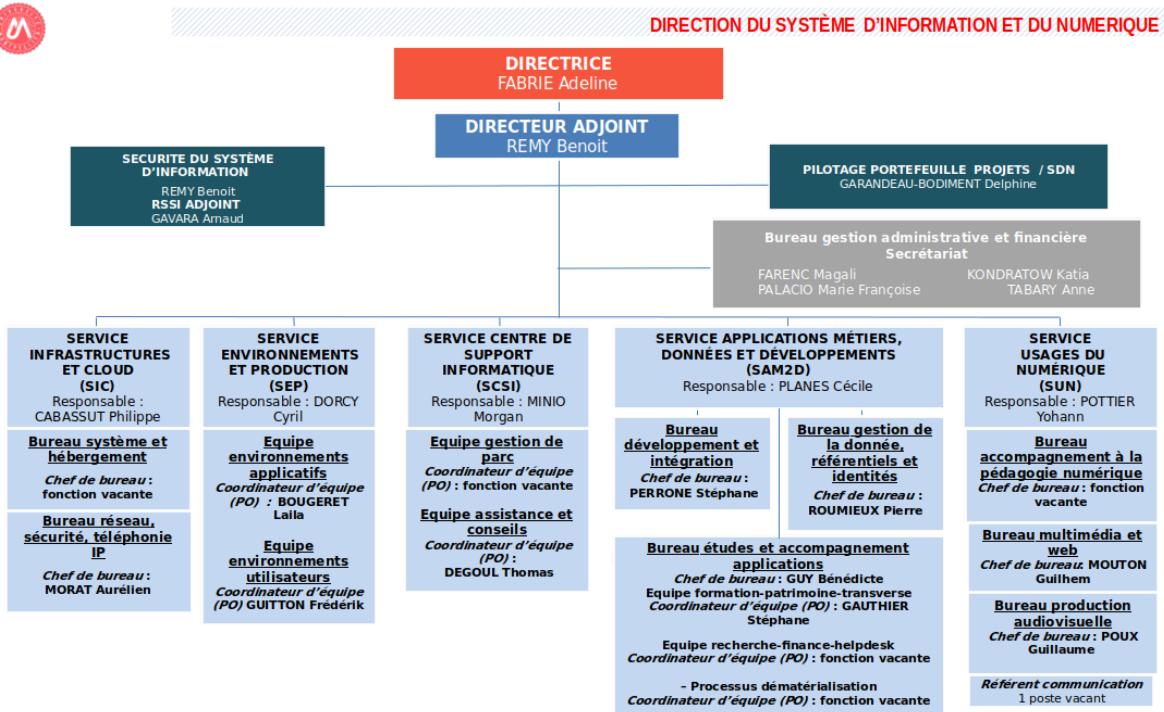


Figure 2 : Organigramme DSIN

1.3 - Le Projet SAPIENS : Un Enjeu Stratégique

SAPIENS* (SAlles Pédagogiques Informatiques d'ENSEignements) représente bien plus qu'un simple outil de gestion : c'est une pierre angulaire dans l'organisation des ressources pédagogiques numériques de l'université. Ce système, initialement développé sous Drupal* 7, a démontré sa valeur mais atteint aujourd'hui ses limites techniques.

L'architecture actuelle, bien que fonctionnelle, présente plusieurs défis :

La fin programmée du support de Drupal 7 pose un risque sécuritaire croissant. L'architecture monolithique* héritée complique l'évolution et la maintenance du système. L'intégration avec des services externes comme Ivanti^[13], critique pour la gestion du parc informatique, nécessite une modernisation pour gagner en performance.

La décision de migrer vers Drupal 10 s'inscrit dans une réflexion plus large sur l'avenir des services numériques universitaires. Cette migration^[21] représente une opportunité de :

- Moderniser l'architecture technique en adoptant les standards actuels
- Améliorer l'expérience utilisateur pour la communauté universitaire

- Optimiser les processus de maintenance et d'évolution du système
- Renforcer la sécurité^[12] et la robustesse de l'infrastructure

Mon intégration dans ce projet intervient à un moment crucial de cette transformation. La complexité technique de la migration, couplée aux enjeux organisationnels, offre un terrain d'apprentissage particulièrement riche pour un étudiant. Cette expérience permet d'appréhender concrètement les défis de la modernisation des systèmes d'information dans un contexte professionnel exigeant.

II - Analyse technique

2.1 - État des lieux de l'infrastructure existante

L'analyse approfondie de l'architecture SAPIENS* existante révèle un écosystème technique complexe, fruit d'années d'évolution et d'adaptations successives. Le système actuel, bien que fonctionnel, présente une architecture qui reflète les paradigmes de développement de Drupal 7, avec ses forces et ses limitations intrinsèques.

Au cœur du système se trouve une architecture monolithique* caractéristique de Drupal 7, où la logique métier se trouve étroitement couplée avec la couche présentation. Cette approche, courante à l'époque du développement initial, pose aujourd'hui des défis en termes de maintenabilité et d'évolutivité. La gestion des salles pédagogiques repose sur un ensemble de scripts personnalisés qui ont été développés pour répondre aux besoins spécifiques de l'université .

L'infrastructure existante s'articule autour de plusieurs composants clés :

- Un système de gestion des contenus basé sur des champs computed* PHP* pour les calculs dynamiques
- Des intégrations complexes avec le service d'inventaire Ivanti^[13] via des webservices*
- Un système de gestion des droits d'accès étroitement lié au CAS^{*[10]} universitaire
- Des modules de visualisation géographique pour la localisation des ressources
- Des scripts de communication avec un serveur de script pour la création des différents services

2.2 - Analyse critique et points d'attention

L'audit technique* réalisé par DropTeam^[5] ainsi que la Mind Map (cf. annexe 1) a mis en lumière plusieurs aspects critiques nécessitant une attention particulière lors de la migration^[2]. L'analyse de ces éléments a permis d'identifier les défis majeurs et d'élaborer une stratégie de migration adaptée.

La complexité de la migration réside notamment dans :

La transformation de l'architecture des services : Un défi majeur réside dans la refonte complète du système de gestion des services utilisateurs. L'obsolescence du module Field Collection^[16] dans Drupal 10 a nécessité une reconception profonde utilisant le système de Paragraphs*, plus moderne et flexible. Cette migration implique notamment :

- La reconstruction du module principal de gestion des services, véritable pierre angulaire du système, qui orchestre désormais la communication entre Drupal et le service wrapper* externe pour la gestion des services MySQL*, PostgreSQL*, Oracle* et Web
- L'implémentation du même système de chiffrement pour sécuriser les données sensibles des services.
- Le développement d'une alternative au module Rules^[17], devenu incompatible, pour maintenir l'automatisation des processus métier*

La gestion des champs calculés. Parallèlement, Drupal 10 a supprimé la possibilité d'exécuter du code PHP* dans les champs pour des raisons de sécurité. Cette évolution majeure nécessite une refonte complète de la logique de calcul, notamment pour :

- Le calcul du nombre de postes par salle
- L'extraction des configurations matérielles
- La gestion des dates de garantie du matériel

L'intégration des services externes. L'interaction avec Ivanti^[18], crucial pour l'inventaire du parc informatique, nécessite une modernisation pour :

- Améliorer la performance des requêtes
- Implémenter un système de cache^[11] efficace
- Garantir la fiabilité des données en temps réel

La gestion des médias. La transition vers le nouveau système de gestion des médias de Drupal 10 implique une refonte de :

- La gestion des images des salles
- Les plans d'aménagement au format SVG*
- Les documents techniques associés

2.3 - Stratégie de migration

Face à la complexité des défis identifiés, notre réflexion s'est orientée vers une approche méthodique et progressive de la migration. La stratégie élaborée s'articule autour de plusieurs dimensions complémentaires, chacune répondant à des enjeux spécifiques tout en s'intégrant dans une vision cohérente du système cible.

L'adoption d'une architecture modulaire* constitue le premier pilier de notre stratégie. En nous appuyant sur les standards Drupal 10 et le framework Symfony*^[4], nous avons conçu une architecture qui privilégie la séparation des responsabilités. Cette approche permet non seulement de simplifier la maintenance future du système, mais également d'assurer une évolutivité maîtrisée des différents composants. La modularité inhérente à cette nouvelle architecture facilite l'intégration de nouvelles fonctionnalités tout en minimisant les impacts sur les modules existants.

La question des performances occupe une place centrale dans notre réflexion. Notre approche intègre des mécanismes d'optimisation sophistiqués, notamment à travers la mise en place d'un système de cache*^[11] intelligent pour les données Ivanti^[13]. Ce système permet de réduire significativement la charge sur les services externes tout en maintenant une fraîcheur satisfaisante des données. L'optimisation des requêtes et des calculs dynamiques s'accompagne d'une refonte des mécanismes d'interaction avec les webservices*, visant à améliorer la réactivité globale de l'application.

L'architecture des services a été repensée dans une optique de durabilité et d'adaptabilité. L'utilisation des Paragraphs*, en remplacement des Field Collections*^[16], s'inscrit dans cette logique en offrant une structure plus flexible et maintenable. Cette nouvelle architecture facilite non seulement la gestion quotidienne du système mais permet également d'envisager sereinement les évolutions futures, qu'il s'agisse d'ajouts de fonctionnalités ou d'intégrations avec de nouveaux services.

La dimension humaine n'a pas été négligée dans notre approche. La transformation progressive du système s'accompagne d'une attention particulière portée à l'expérience utilisateur et à la formation des équipes. Cette

transition, bien que techniquement complexe, doit rester transparente pour les utilisateurs finaux tout en permettant aux équipes techniques de s'approprier les nouvelles technologies et méthodologies mises en œuvre.

III - Réalisation technique

3.1 - Établissement de l'environnement de développement

L'initiation du projet a débuté par une phase cruciale de mise en place de l'environnement de développement*. La découverte et l'adoption de DDev^[15] comme solution de développement local s'est révélée être un choix stratégique majeur. Cet outil, spécialement conçu pour Drupal, offre un environnement conteneurisé qui garantit une reproductibilité parfaite des conditions de développement. Cette approche a permis d'éviter les traditionnels problèmes de "ça fonctionne sur ma machine" tout en facilitant la collaboration avec l'équipe.

L'analyse approfondie de l'audit technique* fourni par DropTeam^[5] a constitué une étape fondamentale dans la compréhension des enjeux. La mind map* (cf. annexe 1) associée a permis de visualiser clairement les interdépendances entre les différents composants du système et d'identifier les points critiques nécessitant une attention particulière. Cette phase d'analyse, bien que chronophage, s'est révélée indispensable pour établir une stratégie de migration^[21] pertinente et anticiper les difficultés potentielles.

La première installation de Drupal 10 en environnement local a servi de terrain d'expérimentation pour appréhender les nouvelles conventions de développement. Cette étape a permis de se familiariser avec les changements majeurs introduits par cette version, notamment l'architecture basée sur Symfony^[41] et les nouvelles pratiques de développement recommandées.

Cette phase d'établissement de l'environnement de développement m'a permis de mobiliser plusieurs apprentissages critiques du référentiel de compétences, notamment AC21.03 (Adopter de bonnes pratiques de conception et de programmation) à travers l'utilisation de DDev pour standardiser l'environnement, ainsi que AC23.02 (Utiliser des serveurs et des services réseaux virtualisés) via la mise en place d'un environnement conteneurisé. L'analyse de l'audit technique a également contribué à AC25.01 (Identifier les processus présents dans une organisation en vue d'améliorer les systèmes

d'information) en me permettant de cartographier les interdépendances du système existant.

3.2 - Migration des structures de données fondamentales

3.2.1 - Architecture des contenus géographiques

La migration des contenus a débuté par la mise en place d'une structure hiérarchique complexe représentant l'organisation spatiale des ressources pédagogiques. Cette hiérarchie, composée de trois niveaux (Campus, Bâtiments, Salles), nécessitait une attention particulière pour maintenir l'intégrité des relations entre les différentes entités.

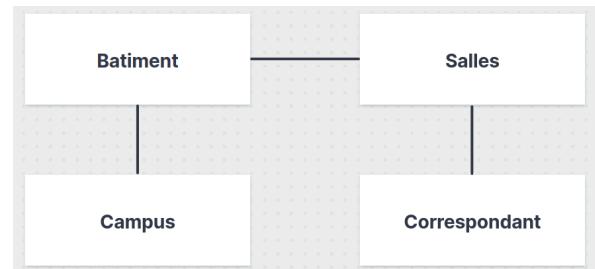
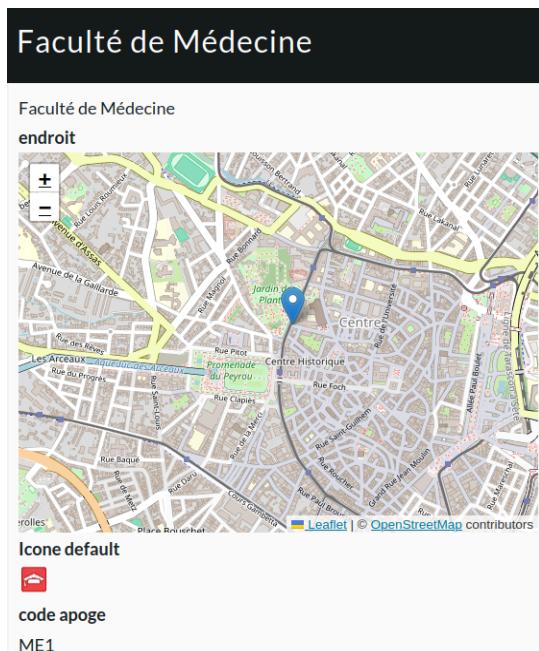


Figure 3 : Structure Contenu Base



La création des types de contenu Campus et Bâtiments s'est déroulée sans difficulté majeure, leurs structures étant relativement simples. Cependant, l'implémentation du système de géolocalisation a nécessité une réflexion approfondie pour assurer une compatibilité optimale avec le module Geofield*^[71] de Drupal 10. Cette intégration permet désormais une visualisation cartographique précise des ressources, améliorant significativement l'expérience utilisateur.

Figure 4 : Exemple Contenu Campus

3.2.2 - Développement du module de gestion des salles

Le véritable défi technique s'est manifesté lors de la configuration des salles. La suppression du support PHP* dans les champs calculés de Drupal 10 a nécessité une refonte complète de la logique de calcul. J'ai donc développé un module personnalisé (cf. annexe 5) intégrant plusieurs innovations techniques :

Système de calcul dynamique des postes

Le module implémente d'un hook*^[14] sophistiqué qui intercepte chaque sauvegarde de salle. Ce hook^[14] déclenche une série d'opérations :

- Extraction et validation de la requête Ivant*i^[13] stockée dans le champ requête inventaire
- Communication asynchrone* avec l'API* Ivanti^[13] pour récupérer les données actualisées
- Analyse et traitement de la réponse JSON* pour calculer le nombre exact de postes
- Mise à jour automatique du champ nombre de postes

Gestion des configurations matérielles

Un système de récupération intelligent des configurations a été mis en place :

- Identification du premier poste de la salle comme référence
- Requête détaillée auprès de l'API* pour obtenir les spécifications techniques
- Mise à jour des champs configurations avec formatage adapté
- Calcul et stockage de la date de fin de garantie pour la planification du renouvellement

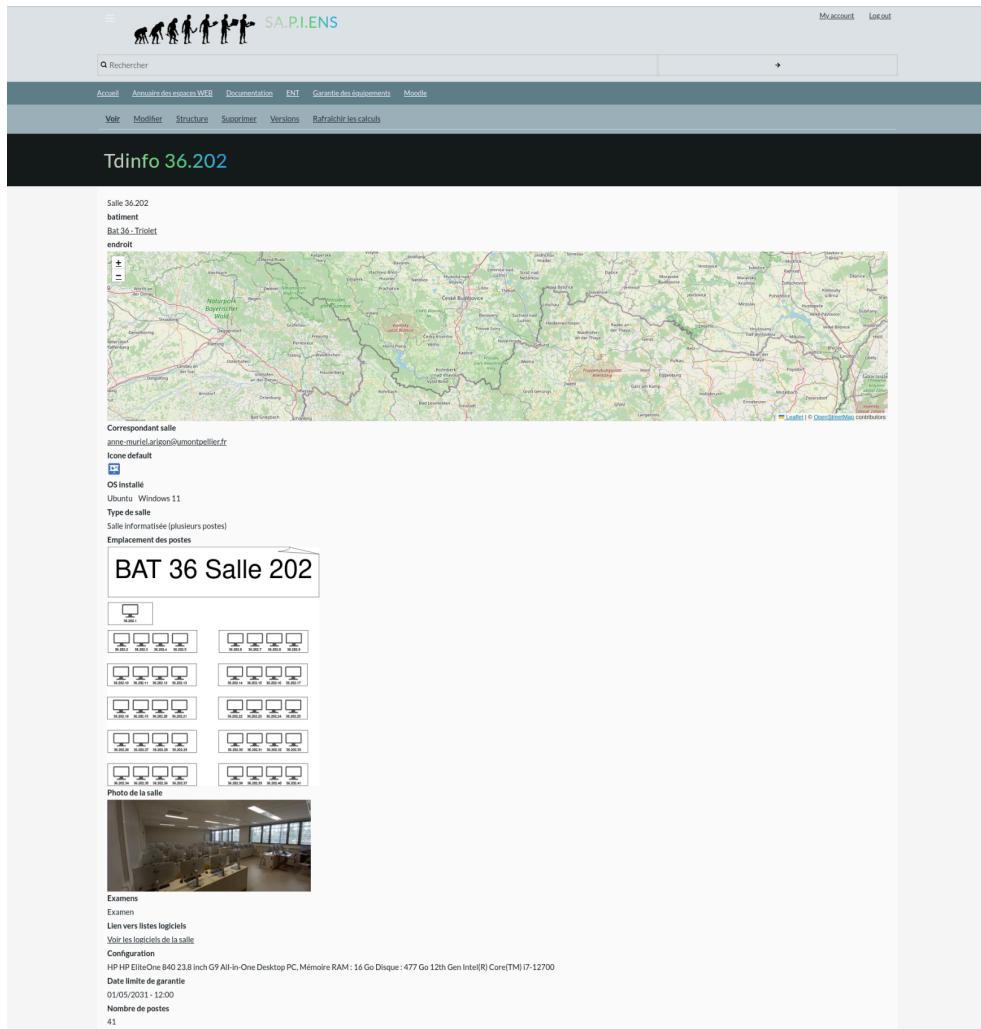


Figure 5 : Exemple contenu salle

3.2.3 - Système de migration automatisée

La migration des données existantes a nécessité le développement d'un système robuste basé sur le module Migrate^[81] de Drupal. J'ai créé un module personnalisé (cf. annexe 12) orchestrant l'ensemble du processus de migration à travers des fichiers YAML* soigneusement structurés. Ce système gère :

Migration géographique

L'un des défis majeurs concernait la migration des données géographiques. J'ai développé une fonction spécifique pour :

- Extraire les coordonnées GPS du format Drupal 7
- Convertir ces données dans le format attendu par Geofield^[71]
- Valider la cohérence des données géographiques
- Générer les mappings appropriés pour la migration

Gestion des médias

La migration^[2] des médias a révélé une incompatibilité fondamentale entre les systèmes de gestion de médias de Drupal 7 et 10. Face à cette difficulté, j'ai adopté une approche hybride :

- Développement d'un script d'extraction des chemins d'accès des images
- Mise en place d'un système de pré-chargement des médias dans Drupal 10
- Configuration du processus de migration pour associer correctement les médias existants

Synchronisation des noms de salles

Pour résoudre les incohérences entre les données Ivanti^[13] et Drupal, j'ai implémenté un système de réconciliation :

- Création d'un hook^{*[14]} spécifique interrogeant l'API Ivanti^[13]
- Comparaison et normalisation* des noms de salles
- Mise en place d'un système de mapping dynamique*
- Journalisation* des modifications pour audit*

3.2.4 - Système de gestion des logiciels et optimisation des performances

L'implémentation du système de gestion des logiciels constituait un défi majeur, nécessitant une refonte complète de l'approche existante. La problématique principale résidait dans l'optimisation des performances de recherche, un point critique identifié dans l'ancienne version de SAPIENS. L'analyse des besoins utilisateurs a révélé la nécessité d'un système permettant une recherche rapide et précise des logiciels installés (cf. annexe 10), tout en offrant une vue contextuelle de leur déploiement* dans les différentes salles.

Innovation dans la gestion du cache^[11]

La réflexion sur les performances a conduit au développement d'un système de cache^{*[11]} sophistiqué. Cette approche novatrice repose sur plusieurs mécanismes complémentaires :

L'implémentation d'un service de mise en cache^{*[11]} quotidienne des données exploite la puissance du système Cron* de Drupal. Chaque nuit, le système effectue une collecte exhaustive des informations sur les logiciels installés,

structurant ces données dans un format optimisé pour les requêtes. Cette approche résout élégamment le problème de latence observé dans l'ancienne version, où chaque recherche déclenchaît un parcours complet des données.

Le système intègre également un mécanisme de rafraîchissement intelligent qui ne met à jour que les données ayant effectivement changé, réduisant ainsi la charge sur les services externes tout en maintenant la fraîcheur des informations. Cette optimisation s'est révélée particulièrement efficace dans le contexte universitaire, où les modifications de configuration logicielle suivent généralement des cycles prévisibles.

Liste des logiciels

Nom du périphérique	Nom logiciel	Version	Salle	Appareil minimum par salle
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
<input type="button" value="Appliquer"/> <input type="button" value="Réinitialiser"/>				
Nom du périphérique	Nom logiciel	Version	Salle	
ADM1004	Adobe Acrobat Reader - Français	24.002.20759	Portable Prêt Examen	
ADM1004	Google Chrome	131.0.6778.139	Portable Prêt Examen	
ADM1004	Ivanti EPM AMT Engine	11.0.5.2696	Portable Prêt Examen	
ADM1004	Ivanti EPM Base Engine	11.0.5.2696	Portable Prêt Examen	

Figure 6 : Vue Liste de logiciel par salle

Cette approche d'optimisation reflète directement l'apprentissage critique* AC22.01 (Choisir des structures de données complexes adaptées au problème) à travers la conception d'un système de cache à plusieurs niveaux. La mise en œuvre de l'algorithme de Levenshtein pour la recherche tolérante aux erreurs s'inscrit parfaitement dans AC22.02 (Utiliser des techniques algorithmiques adaptées pour des problèmes complexes). Quant à l'organisation des données pour faciliter les requêtes, elle mobilise AC24.03 (Organiser la restitution de données à travers la programmation et la visualisation).

Intégration multi-sources

La gestion des logiciels Windows et Linux a nécessité une approche différenciée mais cohérente. Le module développé (cf. annexe 4) orchestre la synchronisation depuis deux sources distinctes :

Pour les logiciels Windows, le système exploite les données déjà présentes dans le cache^[11] Drupal, enrichies par les informations collectées via l'API Ivanti^[13].

Cette intégration permet une vue temps réel du parc logiciel Windows, incluant les versions et les détails de déploiement*.

L'intégration des logiciels Linux a nécessité le développement d'une interface spécifique avec un nouveau webservice*. Cette extension du système démontre sa flexibilité et sa capacité à évoluer pour intégrer de nouvelles sources de données.

Amélioration de l'expérience utilisateur

L'implémentation d'un système de recherche avancé utilisant l'algorithme de Levenshtein* représente une innovation significative dans l'expérience utilisateur. Cette approche tolérante aux erreurs permet aux utilisateurs de trouver rapidement les logiciels recherchés, même en cas d'inexactitudes dans la saisie. Le système calcule dynamiquement la distance entre la chaîne de recherche et les noms de logiciels, offrant des résultats pertinents même en cas de fautes de frappe ou d'orthographe approximative.

3.2.5 - Développement de l'annuaire des espaces web

L'un des objectifs majeurs de cette migration concernait l'amélioration et la modernisation de l'annuaire des espaces web, outil critique permettant aux enseignants de retrouver facilement les espaces web de leurs étudiants à des fins pédagogiques. Ce développement a nécessité une approche méthodique structurée en plusieurs composants complémentaires.

Enrichissement du modèle de données utilisateur

La première étape a consisté en l'enrichissement du modèle de données utilisateur avec l'ajout de champs essentiels :

- **field_codeine** : Identifiant unique dans le système d'information de l'université
- **field_diplome** : Formation suivie par l'étudiant
- **field_etape** : Étape spécifique dans le parcours de formation
- **field_inscription_apogee*** : Référence d'entité* vers les UE auxquelles l'étudiant est inscrit

L'implémentation de ces champs a permis d'établir une structure de données robuste pour supporter les fonctionnalités avancées de recherche et de filtrage nécessaires à l'annuaire.

Module de synchronisation LDAP* (ldap-user-sync)

Un défi majeur résidait dans la nécessité de maintenir synchronisées les données utilisateurs avec le système d'information central de l'université. J'ai développé le module *ldap user sync* (cf. annexe 8) qui répond spécifiquement à cette problématique en offrant une alternative flexible au module LDAP* standard de Drupal.

Ce développement a été motivé par une contrainte architecturale importante : la gestion de l'authentification était déjà assurée par le module CAS, considéré comme plus sécurisé et mieux adapté au contexte universitaire. Le module standard LDAP de Drupal exigeant de prendre en charge l'authentification, une solution sur mesure s'imposait.

Le module implémente les fonctionnalités suivantes :

- Synchronisation automatique des attributs LDAP (*diplome*, *etape*, *code INE*) vers les champs Drupal correspondants à chaque connexion utilisateur
- Interface d'administration permettant de synchroniser manuellement l'ensemble des utilisateurs
- Système de journalisation des opérations de synchronisation
- Configuration flexible du serveur LDAP (anonyme ou authentifiée)

The screenshot shows the 'Synchronisation LDAP' configuration page. It is a multi-step form with the following sections:

- Paramètres de connexion LDAP**:
 - Serveur LDAP**: ldap-app-prep.umontpellier.fr
 - Port LDAP**: 389
 - Base DN**: dc=umontpellier,dc=fr
 - Utiliser la connexion anonyme
- Mapping des attributs LDAP**:
 - Les attributs LDAP suivants seront synchronisés avec les champs utilisateur correspondants.
 - umdiplome → field_diplome
 - umetape → field_etape
 - uidnumber → field_codeine
- Paramètres de l'Inscription Apogée**:
 - Activer la synchronisation des inscriptions Apogée
 - URL de base de l'API Apogée: https://dsitoolsdev3.umontpellier.fr/MODULES/Webservices/
 - Clé d'API Apogée*: BE43C26117C3FAB78FF7AC31AFD641EC

At the bottom are three buttons: **Tester la connexion LDAP**, **Synchroniser tous les utilisateurs**, and **Enregistrer la configuration**.

Figure 7 : Page synchronisation LDAP

Par la suite, j'ai enrichi ce module pour intégrer également la synchronisation du champ *field_inscription_apogee*, qui nécessitait une approche différente. Cette synchronisation s'appuie sur un webservice* sécurisé externe, établissant ainsi une connexion entre les données académiques d'Apogée* et le profil utilisateur Drupal.

Module de synchronisation des UE (sync_ue)

Pour alimenter le champ *field_inscription_apogee*, j'ai développé le module *sync ue* (cf. annexe 3) qui assure la synchronisation automatique des Unités d'Enseignement (UE) depuis le système d'information académique vers Drupal. Ce module :

- Récupère les données des UE via un webservice sécurisé
- Crée automatiquement les nœuds correspondants dans Drupal
- Maintient à jour les informations lorsque des changements surviennent dans le système source
- Établit les relations entre les utilisateurs et les UE auxquelles ils sont inscrits

L'architecture du module repose sur un système de tâches programmées (cron) et d'événements qui permettent une synchronisation régulière tout en limitant la charge sur le système. Une attention particulière a été portée à la gestion des erreurs et à la robustesse du processus de synchronisation dans un environnement où la disponibilité des données est critique.

Paramètres de synchronisation des UE ☆

+ Lancer une synchronisation manuelle

Paramètres de l'API

Les URLs de l'API et les clés sont chargées depuis le fichier `/sites/default/variables.php`.

Environnement à utiliser *

Préproduction

Sélectionnez l'environnement à utiliser pour les appels API (correspond aux clés dans \$fp).

Paramètres de synchronisation

Activer la synchronisation par cron
Exécuter la synchronisation automatiquement lors des exécutions du cron.

Fréquence de synchronisation

Quotidienne

Définit la fréquence à laquelle la synchronisation sera exécutée.

Enregistrer la configuration

Figure 8 : Page synchronisation UE

Moteur de recherche avancé (annuaire_web_search)

La finalisation de l'annuaire des espaces web a nécessité l'implémentation d'un moteur de recherche performant (cf. annexe 13), capable d'interroger efficacement l'ensemble des champs, y compris les références d'entités complexes. Le module *annuaire web* search* répond à ce besoin en offrant :

- Une recherche globale multi-champs (*nom, email, uid, codeine, diplôme, étape, inscription_apogee*)
- Un système d'élimination des doublons dans les résultats
- Une interface de recherche intuitive intégrée à la vue d'annuaire
- Un formatage optimisé des résultats avec des liens directs vers les espaces web

La complexité technique résidait particulièrement dans la gestion des références d'entités, notamment pour le champ *field_inscription_apogee* qui peut contenir de multiples valeurs. Le module implémente une approche sophistiquée avec des jointures SQL dynamiques et une optimisation des requêtes pour maintenir des performances satisfaisantes même avec un volume important de données.

Figure 9 : Page Annuaire des espaces Web

3.2.6 - Modernisation du module de gestion des examens

La transition vers Drupal 10 exigeait une refonte méthodique des interfaces préexistantes, notamment celle dédiée à la gestion des examens universitaires. Le module *exam_bloc_compute* (cf. annexe 9) ne constitue pas une innovation fonctionnelle, mais représente plutôt une réinterprétation architecturale

significative d'une interface précédemment implémentée via des blocs compute PHP dans l'ancienne version de SAPIENS.

Cette migration technique s'inscrivait dans une démarche plus globale de standardisation et de pérennisation du système. L'ancienne représentation des blocs compute PHP, bien que fonctionnelle, présentait des limites en termes de maintenabilité et ne correspondait plus aux bonnes pratiques promues par Drupal 10. J'ai donc entrepris une reconstruction complète selon les conventions contemporaines, transformant ces fragments de code procédural en une architecture modulaire cohérente.

Le défi principal résidait dans la préservation parfaite des fonctionnalités existantes tout en modernisant leur implémentation technique. La communication avec le webservice* externe, déjà établie dans l'ancienne version, a été reconfigurée dans un service dédié suivant le pattern d'injection de dépendances caractéristique de Drupal 10. Cette approche, contrairement à l'implémentation précédente où la logique métier se trouvait directement intégrée dans les blocs d'affichage, introduit une séparation nette entre acquisition des données et présentation.

La gestion des postes informatiques pour les examens constituait un aspect critique du système préexistant que j'ai soigneusement préservé tout en améliorant sa structure. Le contrôle des postes (démarrage, arrêt, vérification du statut) s'effectue désormais à travers une interface standardisée qui maintient la compatibilité avec les scripts externes historiques tout en introduisant une couche d'abstraction facilitant d'éventuelles évolutions futures.

Mes Examens

TP noté en réseau- HAI404ⁱ

03.04.2025 07:00 - 10:00

Salle 36.201, Salle 36.202, Salle 36.203,
Salle 36.204



[Aide](#) ⁱ ou [Voir les archives](#)

Epreuve En salle TD infoⁱ

14.04.2025 07:45 - 09:30

Salle 36.201, Salle 36.202, Salle 36.209



[Aide](#) ⁱ ou [Voir les archives](#)

Figure 10 : Exemple Bloc Examen

L'interface utilisateur, bien que visuellement similaire à sa version antérieure pour maintenir les repères des utilisateurs, bénéficie désormais d'une implémentation rigoureuse via le système de templates Twig*, remplaçant l'ancien rendu généré par du code PHP intégré. Cette approche déclarative améliore considérablement la maintenabilité et la flexibilité de l'interface, tout en préservant l'expérience utilisateur familière.

Une attention particulière a été portée à l'intégration harmonieuse avec les autres composants modernisés du système SAPIENS. Le module s'articule désormais naturellement avec le système d'authentification CAS et interagit de manière standardisée avec le module de gestion des salles, créant ainsi un écosystème cohérent où les différentes fonctionnalités communiquent selon des patterns uniformes.

Cette reconstruction méthodique illustre parfaitement l'approche générale adoptée durant ce projet de migration : non pas une simple transposition fonctionnelle, mais une réinterprétation architecturale respectant les standards contemporains de développement. La gestion des examens s'inscrit désormais dans un cadre technique robuste et évolutif, garantissant sa pérennité face aux futures évolutions de l'écosystème Drupal.

3.3 - Intégration avec l'infrastructure universitaire

Authentification et gestion des utilisateurs

L'intégration avec le système CAS*[\[10\]](#) de l'Université de Montpellier constitue un élément crucial de l'architecture. Le développement s'est concentré sur l'automatisation complète du processus d'inscription et de gestion des utilisateurs. Chaque nouvelle connexion via le CAS[\[10\]](#) déclenche une série d'opérations sophistiquées :

- Création automatique du compte utilisateur dans Drupal
- Récupération et synchronisation des attributs utilisateur (UID, composante, etc.)
- Attribution dynamique des droits et permissions selon le profil

Cette intégration avec le système CAS de l'université a nécessité la mise en œuvre de l'apprentissage critique* AC23.01 (Concevoir et développer des applications communicantes) pour établir le dialogue entre les différents systèmes, ainsi que AC23.03 (Sécuriser les services et données d'un système) pour garantir la sécurité du processus d'authentification. La gestion automatisée des utilisateurs participe également à l'apprentissage critique AC24.02 (Assurer la sécurité des données - intégrité et confidentialité) en implémentant des mécanismes robustes d'attribution des droits.

Système de gestion des services

L'architecture du système de gestion des services constitue l'un des défis majeurs de cette migration. Le passage des Field Collections obsolètes de Drupal 7 vers les Paragraphs de Drupal 10 a nécessité une refonte complète de la structure de données tout en maintenant la compatibilité avec les processus existants. Cette transition complexe s'articule autour de deux modules personnalisés complémentaires développés spécifiquement pour répondre aux besoins particuliers de l'Université.

Le premier module créé, Rules Services (cf. annexe 7), forme la colonne vertébrale du système en orchestrant la communication entre Drupal et les systèmes externes via SSH. Son architecture soigneusement conçue permet de gérer l'ensemble du cycle de vie des services utilisateurs. La création d'un service déclenche un processus sophistiqué qui communique avec les scripts wrapper externes pour provisionner les ressources correspondantes. Ces communications sensibles bénéficient d'un système de chiffrement robuste qui protège notamment les mots de passe, encodés en base64 pour éviter les problèmes liés aux caractères spéciaux durant la transmission.

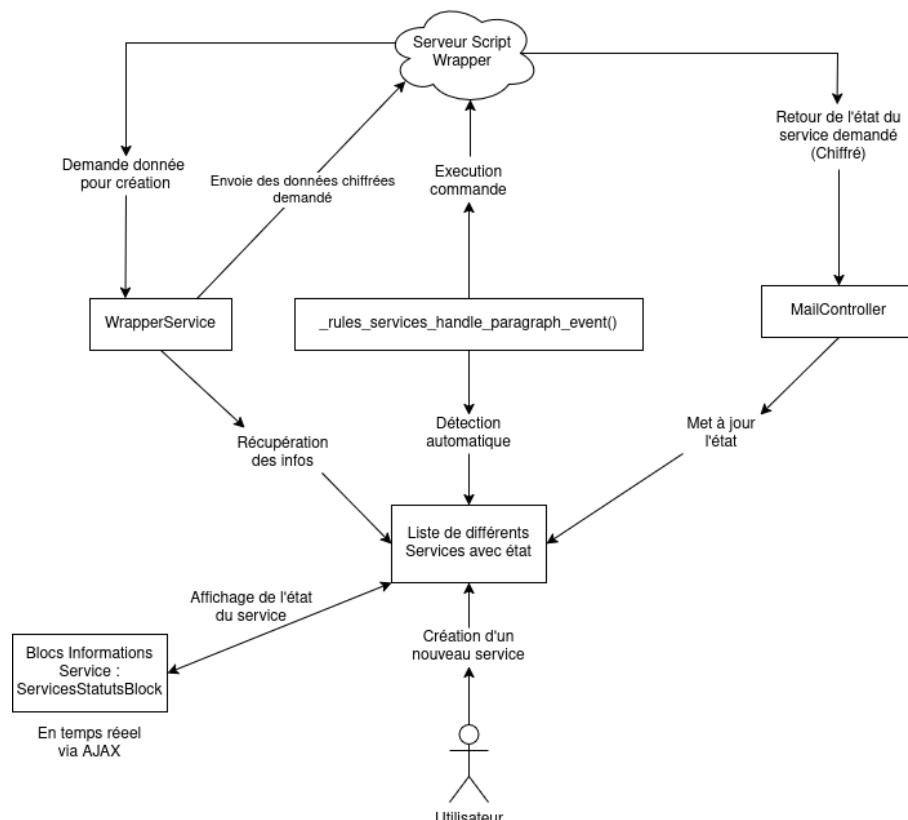


Figure 11 : Architecture de communication service

Le second module créé, Custom Paragraph Actions (cf. annexe 11), vient enrichir cette architecture en introduisant des fonctionnalités avancées de gestion des paragraphes. Il implémente notamment un bouton "Réinitialiser" directement dans l'interface utilisateur, facilitant ainsi les opérations courantes. Ce module gère automatiquement la synchronisation entre les paragraphes de type service et les paragraphes de type BDD correspondants, garantissant l'intégrité des données lors des différentes actions utilisateur.

La gestion des états des services s'effectue désormais à travers un système d'états prédéfinis (add, remove, reset, ok, migrate, newpassword) qui permet un suivi précis du statut de chaque service. Cette approche améliore considérablement la traçabilité des actions et facilite le débogage des problèmes éventuels. Les hooks paragraphs^[6] implementés dans le module interceptent les événements d'insertion, de mise à jour et de suppression pour déclencher les actions appropriées sur les systèmes externes.

Le développement d'une couche d'abstraction entre l'interface utilisateur et les systèmes sous-jacents permet de maintenir la compatibilité avec les scripts existants tout en bénéficiant des avantages techniques offerts par Drupal 10. Cette approche garantit une transition transparente pour les utilisateurs finaux, qui continuent à bénéficier des mêmes fonctionnalités avec une interface améliorée et des performances optimisées.

L'implémentation d'un bloc d'affichage dans le module rules services (cf. annexe 7) de statut en temps réel, avec rafraîchissement automatique via AJAX*, offre aux utilisateurs une visibilité claire sur l'état de leurs services. Ce bloc affiche non seulement le statut de chaque service, mais également des informations pertinentes sur les quotas d'espace disque et web, représentés par des barres de progression colorées selon le niveau d'utilisation.

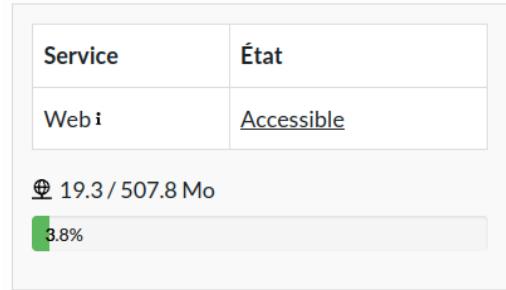


Figure 12 : Bloc affichage service

Le système intègre également une gestion sophistiquée des notifications par email, permettant d'informer automatiquement les utilisateurs lors des changements de statut de leurs services. Une attention particulière a été portée aux notifications spéciales pour les administrateurs des services PostgreSQL*, conformément aux exigences spécifiques de l'Université. Chaque

action est minutieusement journalisée dans un champ dédié, constituant ainsi un historique complet des modifications pour chaque utilisateur.

La prise en compte des différents environnements (production et préproduction) dans l'architecture du module garantit la flexibilité nécessaire au cycle de développement et de test. Le système détermine automatiquement l'environnement en cours d'utilisation et adapte son comportement en conséquence, notamment en ce qui concerne les serveurs wrapper à contacter pour le provisionnement des services.

Cette approche modulaire et extensible prépare efficacement le système pour les évolutions futures, tout en garantissant la robustesse et la sécurité nécessaires à une application critique dans le contexte universitaire.

3.4 - Déploiement et documentation

La phase finale du projet a consisté en le déploiement* sur l'environnement de préproduction. Cette étape a permis de valider l'ensemble des développements dans un contexte proche de la production. L'accès au WikiDSIN* a facilité la documentation exhaustive* du système, assurant ainsi sa pérennité et sa maintenabilité.

La documentation technique produite couvre l'ensemble des aspects du système :

- Architecture générale et choix techniques
- Guides d'installation et de configuration
- Documentation des API* et des interfaces
- Procédures de maintenance et de mise à jour

Cette documentation constitue un élément crucial pour la transmission des connaissances et la maintenance future du système.

3.5 - Développement post-migration : système de suivi des garanties matérielles

Au-delà des objectifs initiaux de migration, le projet a évolué pour intégrer des fonctionnalités amélioratives répondant aux besoins exprimés par les utilisateurs. L'une des réalisations majeures dans ce cadre a été le développement d'un système de suivi des garanties matérielles, incarné par la création du module *view_garantie_correspondant* (cf. annexe 2).

Garantie des équipements

▼ Filtres

Nom du périphérique

Nom de la salle

Nom du bâtiment

Campus

- Tous -

Année d'expiration

- Toutes -

Mois d'expiration

- Tous -

Dernière actualisation des données : 24/03/2025 12:29:59
Nombre d'équipements trouvés : 12
[Afficher tous les résultats](#) [Télécharger en CSV](#)

Nom du périphérique	Salle	Bâtiment	Campus	Date d'expiration de la garantie	Correspondant
PHA0052	Amphi A	Bat A - Pharmacie	Non défini	23/10/2024 - 00:00	assef.yahyaoui@umontpellier.fr
PHA0055	I-3-TD.09 // 3ème ét. - 60 pl	Bat A - Pharmacie	Non défini	23/10/2024 - 23:59	assef.yahyaoui@umontpellier.fr
PHA0051	I-3-TD.08 // 3ème ét. - 60 pl	Bat A - Pharmacie	Non défini	23/10/2024 - 23:59	assef.yahyaoui@umontpellier.fr

Figure 13 : Page Garantie des équipements

Contexte et objectifs

Cette fonctionnalité répond à un besoin opérationnel critique : permettre aux correspondants informatiques de suivre efficacement l'état des garanties matérielles des équipements dont ils ont la responsabilité. L'objectif était d'offrir une vue synthétique et personnalisée, permettant d'anticiper les renouvellements nécessaires et d'optimiser la gestion du parc informatique.

Architecture du module `view_garantie_correspondant`

Le module `view_garantie_correspondant` (cf. annexe 2) s'articule autour d'une architecture sophistiquée qui associe plusieurs composants :

- **Système de rôles et permissions** : Attribution automatique du rôle "correspondant" aux utilisateurs référencés comme responsables de salles
- **Interfaçage avec le système d'inventaire** : Récupération des données de garantie depuis le webservice externe (CSITools*)
- **Cache multi-niveaux¹¹¹** : Implémentation d'un système de cache par salle et par utilisateur pour optimiser les performances
- **Interface de filtrage avancée** : Permettant de filtrer les résultats par nom de périphérique, salle, bâtiment, campus, et date d'expiration

- **Système d'export** : Fonctionnalité d'exportation des données au format CSV pour faciliter le reporting

Le module génère dynamiquement un tableau de bord présentant pour chaque correspondant uniquement les équipements dont il a la charge, avec une organisation par date d'expiration de garantie qui met en évidence les équipements nécessitant une attention prioritaire.

Optimisations techniques

Pour gérer efficacement le volume potentiellement important de données, plusieurs optimisations ont été mises en œuvre :

- Système de cache différencié avec des durées de validité adaptées aux différents types d'information
- Rafraîchissement automatique hebdomadaire via le système cron*
- Exécution asynchrone* des requêtes les plus lourdes
- Pagination configurable des résultats

Cette approche garantit des performances optimales même dans le contexte d'une université disposant d'un parc informatique conséquent, tout en maintenant la fraîcheur des données nécessaire à une gestion efficace des équipements.

Intégration au système existant

L'intégration de cette nouvelle fonctionnalité s'est effectuée de manière non intrusive, en s'appuyant sur les structures de données existantes du système SAPIENS. Le module exploite notamment :

- Les contenus de type "Salle" et leurs métadonnées
- La hiérarchie Campus > Bâtiment > Salle déjà établie
- Le système d'attribution des correspondants implémenté dans le module de gestion des salles

Le développement de ce système post-migration m'a permis d'appliquer concrètement l'apprentissage critique* AC21.01 (Élaborer et implémenter les spécifications fonctionnelles et non fonctionnelles à partir des exigences) en traduisant un besoin opérationnel en solution technique. L'implémentation d'une interface de filtrage avancée répond à AC21.02 (Appliquer des principes d'accessibilité et d'ergonomie), tandis que la gestion des données hétérogènes

issues de différents systèmes illustre parfaitement AC24.04 (Manipuler des données hétérogènes).

IV - Méthodologie et gestion de projet

4.1 - Cadre méthodologique et adaptation de Scrum

L'envergure et la complexité du projet de migration SAPIENS nécessitaient une approche méthodologique alliant rigueur et adaptabilité. La méthodologie Scrum* s'est naturellement imposée comme cadre de référence, tout en exigeant une adaptation fine à notre contexte spécifique de migration système. En tant que Product Owner*, j'ai orchestré la transformation des exigences fonctionnelles en éléments de Product Backlog* structurés et priorisés, naviguant constamment entre fidélité conceptuelle et innovation technique.

La définition et le raffinement du Product Backlog ont constitué un véritable exercice d'équilibrisme intellectuel, nécessitant une compréhension approfondie tant des dimensions techniques que des besoins métiers. L'élaboration méticuleuse de chaque User Story* a permis de maintenir une perspective claire sur l'ensemble des développements tout en garantissant leur ancrage dans les réalités opérationnelles de l'institution. Ce processus de décomposition fonctionnelle, loin d'être purement mécanique, a révélé toute la complexité sous-jacente d'un système d'information universitaire aux multiples ramifications.

L'organisation des sprints* reflétait une recherche constante d'équilibre entre vitesse* productive et maîtrise de la dette technique*. La périodicité bihebdomadaire s'est révélée particulièrement adaptée à notre contexte, offrant suffisamment d'espace pour développer des incrémentations significatives tout en maintenant une flexibilité essentielle face aux contraintes émergentes. Les rituels Scrum* traditionnels ont été judicieusement adaptés à notre configuration particulière de binôme développeur-tuteur, transformant notamment les daily scrums* en sessions de synchronisation bi-hebdomadaires* plus substantielles et constructives.

Cette adaptation méthodologique m'a permis de mettre en pratique l'apprentissage critique* AC25.04 (Définir et mettre en œuvre une démarche de suivi de projet) à travers l'ajustement des rituels Scrum à notre configuration particulière. La transformation des exigences en éléments de Product Backlog

s'inscrit directement dans AC25.02 (Formaliser les besoins du client et de l'utilisateur).

4.2 - Management visuel et pilotage du projet

L'utilisation de Tuleap*[9] comme plateforme centrale de gestion de projet s'est avérée déterminante dans notre démarche. Cet outil, spécifiquement déployé au sein de la DSIN*, nous a permis d'implémenter rigoureusement les pratiques Scrum tout en bénéficiant d'une visibilité panoramique sur l'avancement du projet. La malléabilité de l'interface nous a permis de personnaliser le flux de travail pour refléter précisément notre processus de développement, offrant ainsi une traçabilité complète du cycle de vie des fonctionnalités.

Figure 14 : Outil de suivi Tuleap

Le burndown chart*, actualisé à l'issue de chaque sprint*, constituait un indicateur précieux de notre rythme d'avancement, permettant d'ajuster dynamiquement nos engagements au fil des itérations. Cette métrique quantitative, enrichie par les réflexions qualitatives issues des rétrospectives*, a considérablement facilité l'identification des frictions méthodologiques et l'optimisation progressive de notre processus de développement.

4.3 - Système de gestion de version et intégration continue

L'utilisation rigoureuse de Git comme système de gestion de version a constitué un pilier fondamental de notre approche technique. Mon parcours

s'est caractérisé par une évolution significative des pratiques de versionnement : initialement, j'ai établi mon environnement de développement sur le référentiel Git personnel de l'IUT de Montpellier, créant ainsi un espace de travail isolé propice à l'expérimentation et au prototypage.

La maturation du projet a nécessité une transition stratégique vers le dépôt Git institutionnel de l'Université de Montpellier (cf. annexe 14), accessible publiquement. Cette migration n'était pas qu'une simple question technique, mais répondait à un impératif organisationnel : l'intégration automatisée de la documentation au WikiDSIN. Ce changement illustre parfaitement la dimension évolutive de notre méthodologie, capable de s'adapter aux contraintes émergentes tout en maintenant une cohérence globale dans la gestion des sources.

The screenshot shows a Git repository interface for a project named 'Sap D10'. The left sidebar lists project categories: Gestion, Programmation, Code, Compilation, Déploiement, Opération, Surveillance, and Analyse. The main area displays a commit history for the 'master' branch, with the most recent commit being 'feat(role_field_visibility): Ajout module permettant de gérer la visibilité...' by 'Lambert Raphael' on February 05, 2025. To the right, a table provides validation results for various files and folders, showing the last validation date and the last update date. The table includes rows for 'modules', 'profiles', 'sites', 'themes', '.csslintrc', '.editorconfig', '.eslintignore', '.eslintrc.json', '.gitattributes', '.gitignore', '.ht.router.php', and '.htaccess'. The interface also includes sections for 'Informations sur le projet' (57 validations, 2 branches, 0 étiquette) and 'Date de création' (February 05, 2025).

Nom	Dernière validation	Dernière mise à jour
modules	feat(role_field_visibility): Ajout module p...	Il y a 1 jour
profiles	reload .gitignore	Il y a 1 mois
sites	reload .gitignore	Il y a 1 mois
themes	reload .gitignore	Il y a 1 mois
.csslintrc	reload .gitignore	Il y a 1 mois
.editorconfig	Clone Repo from Gitlab iutInfo	Il y a 1 mois
.eslintignore	reload .gitignore	Il y a 1 mois
.eslintrc.json	reload .gitignore	Il y a 1 mois
.gitattributes	Clone Repo from Gitlab iutInfo	Il y a 1 mois
.gitignore	reload .gitignore	Il y a 1 mois
.ht.router.php	reload .gitignore	Il y a 1 mois
.htaccess	reload .gitignore	Il y a 1 mois

Figure 15 : Environnement GIT

L'adoption d'une discipline rigoureuse de versionnement à travers Git a constitué un pilier important de mon approche technique. Bien que travaillant seul sur le dépôt, j'ai veillé à maintenir une historique claire et cohérente en effectuant systématiquement des commits clairs* et bien documentés pour chaque fonctionnalité développée. Cette pratique méthodique de versionnement a facilité le suivi de l'évolution du code, la détection des problèmes potentiels et la documentation implicite du développement. Cette approche structurée a significativement contribué à la qualité du code produit et à la fluidité du processus de développement, même dans un contexte de développement individuel.

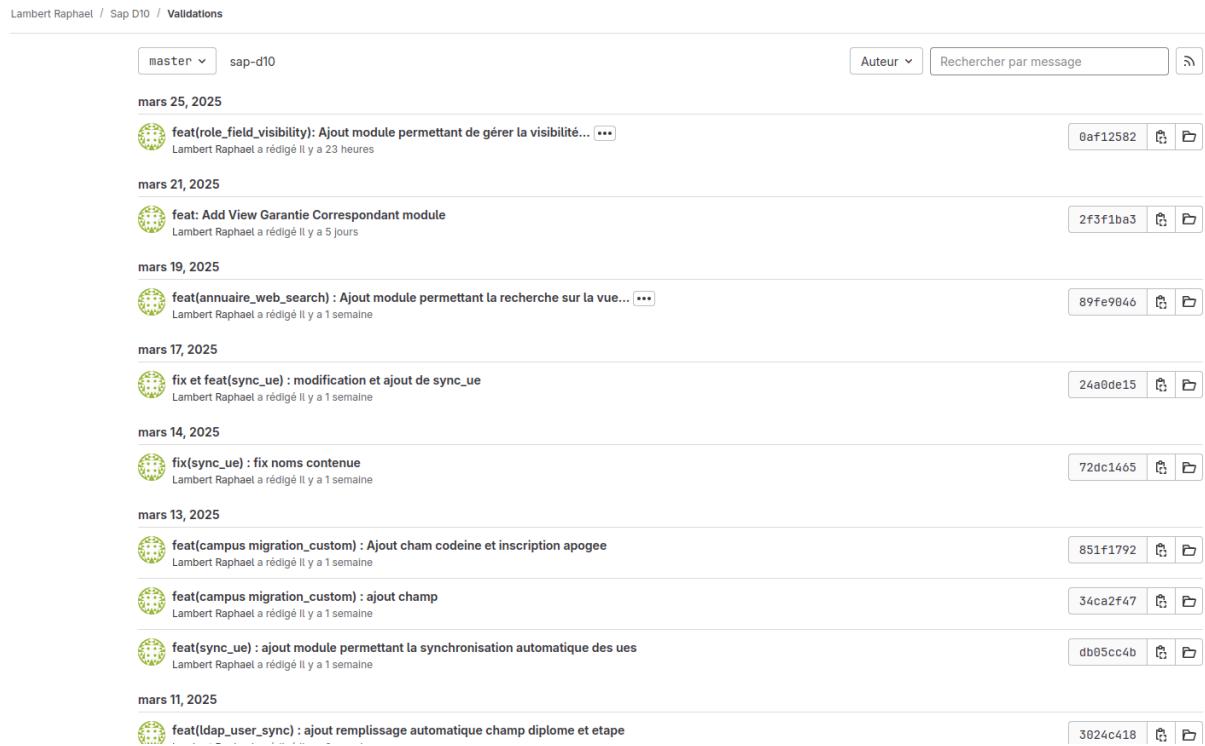


Figure 15 : Commit GIT

4.4 - Collaboration inter-équipes et gestion des dépendances

Un aspect particulièrement enrichissant de ce projet résidait dans sa dimension collaborative transversale. J'ai utilisé pendant toute la durée de mon stage l'outil Teams pour communiquer avec les différents membres de l'équipe et de la DSIN.

La migration du système SAPIENS, loin d'être un exercice purement logiciel, a nécessité une orchestration minutieuse avec différentes équipes techniques de l'université. J'ai ainsi développé une collaboration étroite et substantielle avec l'équipe réseau, collaboration indispensable pour établir les multiples connexions sécurisées entre les différentes composantes de l'infrastructure.

Cette dimension collaborative s'est manifestée concrètement dans l'ouverture et la configuration des canaux de communication entre les divers serveurs impliqués dans l'écosystème SAPIENS : serveurs d'applications, bases de données relationnelles (MySQL* et PostgreSQL*), services web externes et systèmes de script. Chaque connexion nécessitait une analyse précise des flux de données, des protocoles appropriés et des considérations de sécurité,

transformant ainsi cette collaboration en un véritable exercice d'architecture distribuée.

Cette expérience m'a permis d'approfondir ma compréhension des enjeux d'intégration système à grande échelle, dépassant largement le cadre du développement logiciel conventionnel. J'ai notamment saisi l'importance cruciale d'une communication précise et d'une documentation exhaustive lors d'interventions impliquant plusieurs équipes aux expertises complémentaires.

Cette dimension collaborative a été l'occasion de mobiliser les apprentissages critiques liés à l'intégration professionnelle, notamment AC26.01 (Comprendre la diversité, la structure et la dimension de l'informatique dans une organisation) à travers l'interaction avec différentes équipes techniques, et AC26.03 (Mobiliser les compétences interpersonnelles pour travailler dans une équipe informatique) lors des échanges techniques avec les équipes réseau. L'expérience acquise dans la documentation des interventions multi-équipes répond directement à AC26.04 (Rendre compte de son activité professionnelle).

4.5 - Démarche qualité et amélioration continue

L'intégration d'une démarche qualité robuste s'est manifestée à travers plusieurs dispositifs complémentaires. Les revues de code systématiques avec mon tuteur ont permis de maintenir un niveau d'exigence technique élevé tout en facilitant le transfert de connaissances.

Les rétrospectives* de sprint, moments privilégiés d'introspection et d'amélioration, ont permis d'identifier et d'implémenter de nombreuses améliorations processuelles. L'utilisation du format "Start, Stop, Continue"** a facilité l'émergence d'actions d'amélioration concrètes et leur suivi dans le temps.

L'approche d'amélioration continue a également guidé le développement post-migration, notamment pour les fonctionnalités comme l'annuaire des espaces web et le système de suivi des garanties. La méthodologie employée s'est enrichie d'une dimension collaborative accrue, intégrant directement les retours des utilisateurs finaux dans l'évolution des modules. Cette démarche s'est matérialisée par :

- L'organisation de sessions de démonstration régulières auprès des enseignants pour l'annuaire des espaces web

- La mise en place d'un système de feedback simplifié pour les correspondants informatiques
- L'analyse des journaux d'utilisation pour identifier les schémas d'usage et optimiser l'expérience utilisateur

Cette approche a permis d'affiner continuellement les fonctionnalités développées, assurant leur pertinence vis-à-vis des besoins réels des utilisateurs tout en maintenant un haut niveau de qualité technique.

4.6 - Documentation et knowledge management*

La documentation du projet s'est articulée exclusivement autour du WikiDSIN*, plateforme collaborative de la Direction du Système d'Information et du Numérique. Cette centralisation de la documentation sur un unique support s'est révélée particulièrement pertinente, offrant un point d'accès unique à l'ensemble des informations relatives au projet.

Le WikiDSIN a permis de structurer la documentation technique selon plusieurs axes complémentaires. Les pages ont été organisées de manière à couvrir aussi bien les aspects techniques du développement que les procédures de déploiement* et de maintenance. Une attention particulière a été portée à la documentation des modules personnalisés développés durant le projet, incluant leur architecture, leur fonctionnement et les choix techniques effectués.

Cette approche centralisée de la documentation garantit la pérennité du système en facilitant sa maintenance future et son évolution. Les équipes techniques disposent ainsi d'une source unique et fiable d'information, essentielle dans le contexte d'une application critique pour l'université.

V - Perspectives et améliorations futures

L'évolution d'un système d'information universitaire s'inscrit nécessairement dans une dynamique itérative où chaque avancée technique ouvre de nouveaux horizons fonctionnels. Au terme de cette migration fondamentale, plusieurs axes d'amélioration se dessinent avec clarté, porteurs de promesses significatives pour l'écosystème SAPIENS.

5.1 - Intégration native de la procédure d'examen

L'architecture actuelle de gestion des examens, bien que fonctionnellement satisfaisante, repose sur une représentation d'intégration externe qui, à terme, pourrait être avantageusement repensée. La création d'examens s'effectue aujourd'hui par l'intermédiaire d'un processus segmenté : un administrateur SAPIENS doit manuellement configurer l'examen sur une instance Drupal distincte, laquelle transmet ensuite ces paramètres vers un webservice* externe qui orchestre les configurations techniques nécessaires.

Cette médiation systémique, héritée des contraintes historiques du développement, introduit une complexité processuelle qui mérite d'être questionnée. La pertinence d'une intégration native de la procédure d'examen directement au sein de SAPIENS émerge comme une évolution naturelle et prometteuse. Une telle refonte permettrait non seulement de simplifier considérablement le workflow* des administrateurs, mais offrirait également une expérience utilisateur sensiblement plus fluide et cohérente.

La complexité sous-jacente de cette transition réside principalement dans la gestion fine des connexions réseau qu'implique un examen universitaire. En effet, le système actuel permet de configurer sélectivement l'accessibilité des postes d'examen en limitant les connexions externes tout en autorisant précisément certaines destinations spécifiques. Cette granularité de configuration constitue un défi technique substantiel pour une intégration native.

5.2 - Vers une autonomisation des enseignants

Une perspective particulièrement stimulante concerne l'autonomisation progressive du corps enseignant dans la création et la gestion des sessions d'examen. L'approche actuelle, centralisée autour des administrateurs système, pourrait évoluer vers un modèle plus distribué où les enseignants disposeraient d'une interface dédiée leur permettant de configurer eux-mêmes leurs examens.

Cette évolution nécessite cependant une phase intermédiaire d'observation et d'analyse des pratiques. La stratégie envisagée consiste à recueillir méthodiquement les configurations typiques utilisées par les enseignants afin d'établir un catalogue de modèles prédéfinis correspondant aux cas d'usage les plus fréquents. L'analyse de ces patterns d'utilisation permettra de concevoir

des templates intelligents qui encapsuleront les complexités techniques sous-jacentes.

À titre illustratif, l'autorisation d'accès à la plateforme Moodle de l'université ne se résume pas à l'ouverture d'une simple connexion vers un serveur unique, mais implique la configuration d'un ensemble de connexions interconnectées (authentification CAS*, ressources, services annexes). Cette complexité technique, aujourd'hui gérée manuellement par les administrateurs, pourrait être abstraite derrière des configurations prédéfinies intuitives pour les enseignants.

5.3 - Architecture technique envisagée

L'implémentation de cette évolution majeure nécessiterait plusieurs développements structurants :

- Création d'une interface administrative dédiée aux enseignants pour la configuration des examens
- Développement d'un système de templates paramétrable pour les configurations réseau typiques
- Implémentation d'un module de communication directe avec les services réseau de l'université
- Mise en place d'un workflow* d'approbation permettant une validation simplifiée des demandes complexes

Cette transformation architecturale s'inscrit parfaitement dans la continuité de la modernisation, exploitant pleinement les capacités offertes par Drupal 10 en matière de workflows complexes et d'interfaces administratives personnalisées.

L'intégration native de la procédure d'examen représenterait ainsi l'aboutissement logique du processus de modernisation de SAPIENS, unifiant l'ensemble des fonctionnalités critiques au sein d'une plateforme cohérente et autonome, tout en offrant une expérience utilisateur considérablement enrichie pour l'ensemble de la communauté universitaire.

Conclusion

La migration du système SAPIENS de Drupal 7 vers Drupal 10 représentait un défi technique et organisationnel majeur pour la Direction du Système d'Information et du Numérique de l'Université de Montpellier. Au terme de ce stage, les objectifs initiaux ont été pleinement atteints avec la mise en place d'une architecture moderne répondant aux exigences actuelles de sécurité et de performance.

Les principales réalisations comprennent :

- La refonte complète des modules personnalisés en utilisant les standards Drupal 10
- La transition réussie du système Field Collections vers Paragraphs pour la gestion des services
- L'implémentation d'un système de cache innovant optimisant significativement les performances
- Le développement d'un annuaire des espaces web entièrement repensé
- La création d'un système de suivi des garanties matérielles pour les correspondants informatiques

L'adoption des standards Drupal 10 et l'implémentation de bonnes pratiques de développement garantissent la maintenabilité à long terme de l'application. La nouvelle architecture modulaire facilitera l'intégration de futures fonctionnalités, tandis que les mécanismes d'optimisation permettront d'absorber la croissance des besoins.

Acquis techniques et professionnels

Sur le plan technique, ce stage m'a permis de développer une expertise approfondie dans plusieurs domaines, mobilisant de nombreux apprentissages critiques du référentiel de compétences du BUT Informatique :

- La maîtrise des technologies Drupal 10 et de l'écosystème Symfony, notamment à travers la réécriture des modules personnalisés et l'adaptation des hooks au nouveau framework (AC21.03, AC21.01)
- La conception et l'implémentation de systèmes de cache à plusieurs niveaux, une compétence particulièrement précieuse pour l'optimisation des applications web (AC22.01, AC22.02)
- Le développement de mécanismes d'intégration robustes avec des services externes (Ivanti, LDAP, CAS), un enjeu technique majeur dans les systèmes d'information modernes (AC23.01, AC23.03)

La confrontation avec des défis concrets comme la transformation des Field Collections en Paragraphs a nécessité une compréhension approfondie des structures de données et des mécanismes internes de Drupal (AC22.01). J'ai notamment dû résoudre des problèmes complexes liés à la gestion des services utilisateurs, comme le maintien de l'intégrité des données lors des opérations d'ajout, de modification et de suppression des services (AC24.02).

Méthodologie et travail d'équipe

L'adaptation de la méthodologie Scrum au contexte spécifique de ce projet a constitué un apprentissage précieux (AC25.04). En tant que Product Owner, j'ai dû orchestrer la transformation des exigences fonctionnelles en éléments de Product Backlog structurés et priorisés (AC25.02), un exercice d'équilibrisme entre fidélité fonctionnelle et innovation technique.

La collaboration étroite avec mon tuteur de stage m'a permis de bénéficier d'un accompagnement technique de qualité tout en développant mon autonomie (AC26.03). Cette relation de confiance s'est traduite par :

- Des sessions de revue de code constructives qui ont significativement amélioré la qualité du code produit (AC21.04)
- Une autonomie croissante dans la prise de décisions techniques
- Un équilibre entre initiative personnelle et respect des standards de l'équipe (AC26.02)

Cette expérience m'a particulièrement démontré l'importance d'une documentation exhaustive (AC26.04), notamment pour assurer la pérennité des développements dans un contexte universitaire où la transmission des connaissances est primordiale.

Ce stage a finalement confirmé mon intérêt pour les projets techniques complexes nécessitant une approche méthodique et une vision globale. La satisfaction de voir un système critique être modernisé et optimisé, tout en préservant sa compatibilité avec l'existant, constitue une expérience professionnelle particulièrement enrichissante qui guidera mes choix de carrière futurs.

Les apprentissages critiques mobilisés tout au long de ce stage m'ont permis de valider concrètement les six compétences ciblées par le référentiel du BUT Informatique, me préparant efficacement à mon entrée dans le monde professionnel.

Annexes

ANNEXE 1 : MindMap Migration S.A.P.I.E.N.S :

https://drive.google.com/file/d/1sKe99Ta5puWK8dVjQXkLHM0y9qMuVtKb/view?usp=drive_link

ANNEXE 2 : Documentation Module Vue Garantie Correspondant :

https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/view_garantie_correspondant/README.md?ref_type=heads

ANNEXE 3 : Documentation Module de Synchronisation UE :

https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/sync_ue/README.md?ref_type=heads

ANNEXE 4 : Documentation Module Software Sync :

https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/software_sync/README.md?ref_type=heads

ANNEXE 5 : Documentation Module Salle Compute :

https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/salle_compute/README.md?ref_type=heads

ANNEXE 6 : Documentation Module Rules Services Import :

https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/rules_services_import/README.md?ref_type=heads

ANNEXE 7 : Documentation Module Rules Services :

https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/rules_services/README.md?ref_type=heads

ANNEXE 8 : Documentation Module de synchronisation LDAP :

https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/ldap_user_sync/README.md?ref_type=heads

ANNEXE 9 : Documentation Module Exam Bloc Compute :

https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/exam_bloc_compute/README.md?ref_type=heads

ANNEXE 10 : Documentation Module Custom Software View :

https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/custom_software_view/README.md?ref_type=heads

ANNEXE 11 : Documentation Module Custom Paragraph Actions:

https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/custom_paragraph_actions/README.md?ref_type=heads

ANNEXE 12 : Documentation Module Custom Migration:

https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/custom_migration/README.md?ref_type=heads

ANNEXE 13 : Documentation Module recherche de l'annuaire web :

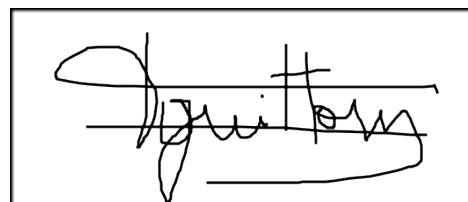
https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10/-/blob/master/modules/custom/annuaire_web_search/README.md?ref_type=heads

ANNEXE 14 : GIT SAPIENS :

<https://gitlab.etu.umontpellier.fr/e20220005734/sap-d10>

Vu par M. Frédéric Guitton,

le 27/03/2025

A handwritten signature in black ink, enclosed in a rectangular border. The signature reads "Frédéric Guitton".